Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 15.04.06 Мехатроника и робототехника
Отделение школы (НОЦ) Отделение автоматизации и робототехники

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Исследование роботизированной мобильной платформы KUKA youBot

УДК 007.52:004.42

Студент

Группа	ФИО	Подпись	Дата
8ЕМ61	Ефименко Анастасия Павловна		

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Руководитель ВКР	Гончаров В.И.	д.т.н.		
Руководитель ООП	Малышенко А.М.	д.т.н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Петухов О.Н.	к.э.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Бородин Ю.В.	к.т.н.		

ДОПУСТИТЬ К ЗАЩИТЕ:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Руководитель ОАР	Леонов С.В.	к.т.н.		

Томск – 2018 г.

Запланированные результаты обучения по программе

Код результата	Результат обучения (выпускник должен обладать)
Общекультурные компетенции	
ОК-1	способностью совершенствовать и развивать свой интеллектуальный и общекультурный уровень
ОК-2	способностью к самостоятельному обучению с помощью современных информационных технологий новым методам исследования, к постоянному обновлению и расширению своих знаний, к изменению в случае необходимости научного и научно-производственного профиля своей профессиональной деятельности
ОК-3	способностью использовать в практической деятельности новые знания и умения, как относящиеся к своему научному направлению, так и в новых областях знаний, непосредственно не связанных с профессиональной сферой деятельности
ОК-4	готовностью использовать на практике приобретенные умения и навыки в организации исследовательских и проектных работ, выполняемых малыми группами исполнителей
Общепрофессиональные компетенции	
ОПК-1	способностью представлять адекватную современному уровню знаний научную картину мира на основе знания основных положений, законов и методов естественных наук и математики
ОПК-2	владением в полной мере физико-математическим аппаратом, необходимым для описания и исследования разрабатываемых систем и устройств

ОПК-3	владением современными информационными технологиями, готовностью применять современные средства автоматизированного проектирования и машинной графики при проектировании систем и их отдельных модулей, знать и соблюдать основные требования информационной безопасности
ОПК-4	готовностью собирать, обрабатывать, анализировать и систематизировать научно-техническую информацию по тематике исследования, использовать достижения отечественной и зарубежной науки, техники и технологии в своей профессиональной деятельности
ОПК-5	способностью использовать методы современной экономической теории при оценке эффективности разрабатываемых и исследуемых систем и устройств, а также результатов своей профессиональной деятельности
ОПК-6	готовностью пользоваться основными методами защиты производственного персонала и населения от возможных последствий аварий, катастроф, стихийных бедствий
Профессиональные компетенции	
ПК-1	способностью составлять математические модели мехатронных и робототехнических систем, их подсистем, включая исполнительные, информационно-сенсорные и управляющие модули, с применением методов формальной логики, методов конечных автоматов, сетей Петри, методов искусственного интеллекта, нечеткой логики, генетических алгоритмов, искусственных нейронных и нейро-нечетких сетей
ПК-2	способностью использовать имеющиеся программные пакеты и, при необходимости, разрабатывать новое

	программное обеспечение, необходимое для обработки информации и управления в мехатронных и робототехнических системах, а также для их проектирования
ПК-3	способностью разрабатывать экспериментальные макеты управляющих, информационных и исполнительных модулей мехатронных и робототехнических систем и проводить их исследование с применением современных информационных технологий
ПК-4	способностью осуществлять анализ научно-технической информации, обобщать отечественный и зарубежный опыт в области мехатроники и робототехники, средств автоматизации и управления, проводить патентный поиск
ПК-5	способностью разрабатывать методики проведения экспериментов и проводить эксперименты на действующих макетах и образцах мехатронных и робототехнических систем и их подсистем, обрабатывать результаты с применением современных информационных технологий и технических средств
ПК-6	готовностью к составлению аналитических обзоров и научно-технических отчетов по результатам выполненной работы, в подготовке публикаций по результатам исследований и разработок
ПК-7	способностью внедрять на практике результаты исследований и разработок, выполненных индивидуально и в составе группы исполнителей, обеспечивать защиту прав на объекты интеллектуальной собственности
ПК-8	готовностью к руководству и участию в подготовке технико-экономического обоснования проектов создания

	мехатронных и робототехнических систем, их подсистем и отдельных модулей
ПК-9	способностью к подготовке технического задания на проектирование мехатронных и робототехнических систем их подсистем и отдельных устройств с использованием стандартных исполнительных и управляющих устройств, средств автоматики, измерительной и вычислительной техники, а также новых устройств и подсистем
ПК-10	способностью участвовать в разработке конструкторской и проектной документации мехатронных и робототехнических систем в соответствии с имеющимися стандартами и техническими условиями
ПК-11	готовностью разрабатывать методику проведения экспериментальных исследований и испытаний мехатронной или робототехнической системы, способностью участвовать в проведении таких испытаний и обработке их результатов
ПК-12	способностью организовывать работу малых групп исполнителей
ПК-13	готовностью разрабатывать техническую документацию (графики работ, инструкции, планы, сметы) по утвержденным формам
ПК-14	готовностью применять методы профилактики производственного травматизма, профессиональных заболеваний, предотвращения экологических нарушений
ПК-15	способностью проводить наладку, регулировку и настройку мехатронных и робототехнических систем различного назначения

ПК-16	готовностью выполнять отладку программно-аппаратных комплексов и их сопряжение с техническими объектами в составе мехатронных и робототехнических систем
ПК-17	готовностью к участию в проведении испытаний и сдаче в эксплуатацию опытных образцов мехатронных и робототехнических систем
ПК-18	готовностью к участию в разработке программ регламентных испытаний, поверке и оценке состояния мехатронных и робототехнических систем различного назначения, а также их отдельных подсистем
ПК-19	способностью провести профилактический контроль технического состояния и функциональную диагностику мехатронных и робототехнических систем различного назначения, а также их отдельных подсистем
ПК-20	способностью составить инструкции по эксплуатации мехатронных и робототехнических систем и их аппаратно-программных средств
ПК-21	готовностью к составлению заявок на оборудование и комплектующие, к участию в подготовке технической документации на ремонт оборудования

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 15.04.06 Мехатроника и робототехника
Отделение школы (НОЦ) Отделение автоматизации и робототехники

УТВЕРЖДАЮ:
Руководитель ООП
_____ _____ Малышенко А.М.
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Магистерской диссертации

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	ФИО
8ЕМ61	Ефименко Анастасии Павловне

Тема работы**:**

Исследование роботизированной мобильной платформы KUKA youBot	
Утверждена приказом директора (дата, номер)	28.05.2018 №3816/с

Срок сдачи студентом выполненной работы:	05.06.18

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Объектом исследования является роботизированная мобильная платформа KUKA youBot на всенаправленных колёсах. Применяется для обучения студентов основам робототехники. Конечным продуктом является программное обеспечение для управления роботом.
Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	1. Управление роботом KUKA youBot в симуляторе; 2. Распознавание объекта: поиск объекта, находящегося на площадке, и получение его координат; 3. Захват объекта с помощью полученных координат и его перенос в нужное местоположение; 4. Движение по заданной траектории;

	5. Исследование свойств KUKA youBot для его применения для наземных испытаний трансформируемых конструкций космических аппаратов.
Перечень графического материала *(с точным указанием обязательных чертежей)*	Презентация

Консультанты по разделам выпускной квалификационной работы
(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент	Петухов О.Н.
Социальная ответственность	Бородин Ю.В.
Иностранный язык	Горбатова Т.Н.

Названия разделов, которые должны быть написаны на русском и иностранном языках:

Литературный обзор

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	10.03.18

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор	Гончаров В.И.	д.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ЕМ61	Ефименко Анастасия Павловна		

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки (специальность) 15.04.06 Мехатроника и робототехника
Уровень образования Магистратура
Отделение школы (НОЦ) Отделение автоматизации и робототехники
Период выполнения Весенний семестр 2018 учебного года

Форма представления работы:

Магистерская диссертация

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	05.06.18

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
20.04.18	Литературный обзор	25
15.05.18	Экспериментальная часть	30
21.05.18	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	20
23.05.18	Социальная ответственность	10
26.05.18	Приложение на иностранном языке	15

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор	Гончаров В.И.	д.т.н.		

СОГЛАСОВАНО:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Профессор	Малышенко А.М.	д.т.н.		

РЕФЕРАТ

Выпускная квалификационная работа 88 с., 4 рис., 12 табл., 10 источников, 2 прил.

Ключевые слова: KUKA youBot, V-REP, OpenCV, Linux, Lua, роботизированная мобильная платформа.

Объектом исследования является роботизированная мобильная платформа KUKA youBot, которая предназначена для решения исследовательских и промышленных задач.

В основу научных исследований данной работы положены методы математического (обзор описания кинематики и динамики робота) и компьютерного моделирования (имитационное моделирование).

Цель работы – изучение кинематики и динамики KUKA youBot, разработка программного обеспечения для решения простых робототехнических задач и обучения студентов основам работы с данным роботом.

В процессе исследования произведён аналитический обзор по данной теме, изучены существующие российские и зарубежные источники, разработано ПО для решения базовых робототехнических задач.

Результаты исследований могут послужить основой для выполнения лабораторных работ студентами младших курсов. Это позволит студентам быстрее изучить основы работы с KUKA youBot и перейти к решению более сложных задач на KUKA youBot.

Определения, обозначения, сокращения, нормативные ссылки

В работе используются следующие обозначения и сокращения:

СУ – система управления;

ПО – программное обеспечение;

МК – микроконтроллер;

ОС – операционная система;

IDE – интегрированная среда разработки (англ. *Integrated Development Invironment*);

ЯП – язык программирования;

ОГЛАВЛЕНИЕ

Введение

Промышленность во всех её отраслях играет очень важную роль. В современном мире большое внимание уделяется цифровизации и роботизации производства. Роботизация является основной составляющей автоматизации производства. На практике этот процесс заключается в применении роботов и роботизированных систем на предприятиях в промышленном масштабе. Актуальными являются задачи по распознаванию объектов и их перемещению; по движению по заданной траектории и т.д. Автоматические линии можно оснастить промышленными роботами, наличие которых позитивно отобразится на функционировании всего комплекса оборудования. Также такие механизмы могут быть включены в гибкие автоматизированные производства. Достоинство промышленных роботов в том, что для их перенастройки на изготовление других изделий не требуется особых затрат, что обеспечивает процессу выпуска продукции достаточную универсальность.

Из сказанного выше, роботы играют важную роль в промышленности, являясь неотъемлемой её частью, поэтому необходимо правильно организовать их работу. KUKA youBot является универсальной роботизированной платформой, применяемой в исследовательских целях студентами и распространённым представителем мобильных роботов на шведских колёсах для выполнения производственных задач.

1. Управление роботом KUKA youBot в симуляторе

Анализ имеющегося программного обеспечения для робота.

В настоящее время существует множество интегрированных сред разработки (IDE) для планирования движения роботов. Формально, с одной стороны, они могут быть классифицированы в функции графического интерфейса: среды визуального программирования и среды программирования на основе текста. В свою очередь, каждая группа включает среды для решения проблем и универсальные IDE (поддерживающие разные модели роботов). Тем не менее, с другой стороны, IDE можно было бы разделить на функциональные возможности и реализацию: образовательные и промышленные IDE. В настоящее время существует множество интегрированных сред разработки (IDE) для планирования движения роботов. Формально, с одной стороны, он может быть классифицирован в функции графического интерфейса: среды визуального программирования и среды программирования на основе текста. Опять же, каждая группа включает среды для решения проблем и универсальные IDE (поддерживающие разные модели роботов). Тем не менее, с другой стороны, IDE можно было бы разделить на функциональные возможности и реализацию: образовательные и промышленные IDE. Например, IDE NXT-G и Trik Studio можно было бы направлять в визуальные образовательные среды IDE. NXT-G - графическая среда разработки, разработанная для популярных настраиваемых программируемых роботов Lego Mindstorms NXT. Он используется для программирования модуля NXT Brick. Это программное обеспечение имеет интуитивно понятный интерфейс. Разработка программы выглядит как временная блок-схема. Программный поток определяет последовательность команд. Trik Studio - это коммерческое программное обеспечение, ориентированное на обучение программированию Trik. Он включает в себя симулятор, который позволяет тестировать программу без реальной системы роботов. Кроме того, есть Microsoft Robotics Developer Studio (MRDS) (Microsoft, 2016). Его можно было бы отнести к полуобразовательной и

промышленной визуализации IDE. Платформа MRDS включает в себя язык Visual Programming (VPL) и визуально смоделированную 3D-среду. VPL рекомендуется для новичков. Он использует стратегии программирования в качестве блок-схемы. MRDS также поддерживает C # для профессиональных разработчиков. Среди образовательных проблемных текстовых сред являются наиболее популярные IDE RobotC и BricxCC. RobotC является лидером среди языков программирования для соревнований роботов. Он основан на языке программирования C, не имеет бесплатной версии. BricxCC - это самый популярный бесплатный программный язык программирования NXC. Есть много блоков для работы с Lego Mindstorms. Кроме того, существует универсальная образовательная текстовая IDE Arduino. Он получил широкое распространение из-за использования Arduino для образования. Arduino GUI содержит текстовую консоль, панель инструментов и меню. Программное обеспечение позволяет передавать данные контроллеру. Сравнительное исследование образовательных IDE представлено в таблице 1. Существует пять самых популярных IDE для промышленного и научного использования: KUKA Sim Pro (KUKA AG, 2017), RobotStudio (ABB, 2017), MotoSim Touch (Yaskawa, 2017), V -REP (Coppelia Robotics GmbH, 2016), RoboDK (RoboDK, 2017). RobotStudio на базе ABB VirtualController. Он позволяет имитировать с высоким уровнем реальности, используя программу реального времени и настраивать модели, а также реальные манипуляторы. Для этого программного обеспечения разработаны специальные плагины (PowerPacs) для различных промышленных процессов: ArcWelding PowerPac, Cutting PowerPac, Painting PowerPac, Паллетирование PowerPac, выбор PowerPac и т. Д. Каждый PowerPac имеет собственный интерфейс. MotoSim Touch - эффективная автономная среда разработки. Он разработан специально для университетов и исследовательских центров. Эта IDE содержит систему программирования MotoSim Touch и виртуальный роботизированный контроллер MotoSim EG-VRC. MotoSim Touch может переключаться между двумя режимами: программным (виртуальным) и аппаратным. Каждый из

режимов использует MotoSim-EG-VRC для обучения автономному программированию и виртуального моделирования. KUKA Sim Pro - программное обеспечение для 3D-моделирования с KUKA Robotics. Это специальное программное обеспечение для исследовательских центров и системных интеграторов. Это официальное программное обеспечение немецкой компании KUKA AG. В стандартной библиотеке KUKA Sim Pro есть много компонентов. Интеллектуальные компоненты имеют точки захвата, которые позволяют подключать их между собой. Возможны также подключения с аналоговыми и цифровыми сигналами KUKA.OfficeLite. Существует открытый язык сценариев Python 2.6 для программирования процессов пользователя. KUKA Sim Pro поддерживает API и COM для разработки пользовательских плагинов. RoboDK - мощный инструмент для автономного программирования и моделирования сложных процессов. Разработчиком программного обеспечения является канадская компания École de Technologie Supérieure (ETS). Это программное обеспечение позволяет разрабатывать и моделировать стратегию поведения роботов разных производителей с языком Python. Кроме того, существует возможность разработки программ со специальными языками для разных моделей: например, RAPID - для робототехники ABB, KRL - для робототехники KUKA и так далее. Последней из изученных программ является V-REP. У нее есть бесплатная версия для образования. Разработчиком программного обеспечения является шведская компания Coppelia Robotics. Программное обеспечение, разработанное для быстрого моделирования процессов, проверки, удаленного мониторинга роботизированных систем. Он имеет распределенную архитектуру управления: каждый объект / модель может управляться индивидуально с помощью собственного сценария (сценария), плагина, узла операционной системы робота (ROS), API удаленного клиента. V-REP - идеальное решение для разработки систем управления несколькими роботами.

Таблица 1. Сравнительный анализ образовательных IDE

Образовательная IDE	Преимущества	Недостатки	Сфера применения
NXT-G	Простота использования; ясность интерфейса	Определение переменных не является быстрым и легким;	Позволяет создавать простые программы; применима для школьников
Microsoft Robotics Developer Studio	Высокоуровневое программирование; включает симулятор, который позволяет проверять программу без реальной робототехнической системы; поддерживает широко распространенные роботизированные системы	Определение переменных не является быстрым и легким; не может загружать программу на робот (связь только с Bluetooth)	Программное обеспечение может использоваться в академических и коммерческих целях
Trik Studio	Включает симулятор, который позволяет проверять программу без реальной	Громоздкость; ориентированность только для одного робота;	Используется для программирования роботов Trik

	робототехнической системы		
RobotC	На основе языка программирования C; без возрастных ограничений; есть много бесплатных уроков	Коммерческое программное обеспечение	Для соревнований роботов
BricxCC	Дружественный интерфейс; комбинация команд высокого уровня и низкого уровня; точность управления	Отсутствие визуализации	Используется для программирования Lego
Arduino	Поддержка NXC; точность контроля	Не поддерживает Lego; ориентирован только на платформу Arduino; необходимо знать микроэлектронику	Программирование роботов Arduino

Таблица 2. Сравнительный анализ промышленных IDE

Свойство ПО/IDE	V-REP	RoboDK	Kuka Sim Pro	RobotStudio	MotoSim Touch
Свободная версия	+	+	-	+	+
Большой объём продукта	+	+	+	+	+
Поддержка различных производителей	+	+	-	-	-
Кросс-платформенное программное обеспечение	+	+	+	+	+
Поддержка 3D-анимации в PDF	-	-	+	-	-
API и добавочные библиотеки	+	+	+	+	+
Версия клиент-сервер	-	-	-	-	-
Основной язык программирования	Lua	Python	Python	Rapid	Arduino

Работа в симуляторе V-REP

V-REP — гибкая и масштабируемая платформа для робомоделирования. Главной целью симулятора является предоставления большого количества инструментов и возможностей для симуляции. С этим есть сложности, ведь разнообразие робототехнических систем не позволяет заранее предвидеть специфику симулятора. Кроме того, некоторым

пользователям нужен гибкий подход, который даст возможность работать с простым языком программирования, а также позволит сделать симулятор портативным, подходящим для всех типов моделей роботов, и самое главное масштабируемым. Отличительной особенностью V-REP являются встроенные скрипты. Основной цикл моделирования lua скрипт (так называемый “основной сценарий”) — обрабатывает общие функциональные возможности. Например, он вызывает разные функции для обработки кинематики или динамики. Основной сценарий также отвечает за вызов дочернего скрипта каскадным способом. Дочерний скрипт, в отличие от основного скрипта, прикрепляется к конкретному объекту или конкретной части моделирования в процессе цикла моделирования. Он является неотъемлемой частью сценария объекта, и будет повторяться вместе с ним. Как таковой дочерний скрипт представляет собой вполне портативный и масштабируемый элемент управления: в нем есть один единый файл, содержащий определение модели вместе с ее контролем или функционалом, нет проблемы совместимости на разных платформах, нет необходимости в явной компиляции, никакого конфликта между несколькими версиями одной и той же модели и др. Дочерние скрипты могут быть запущены в потоковой или не потоковой реализации.

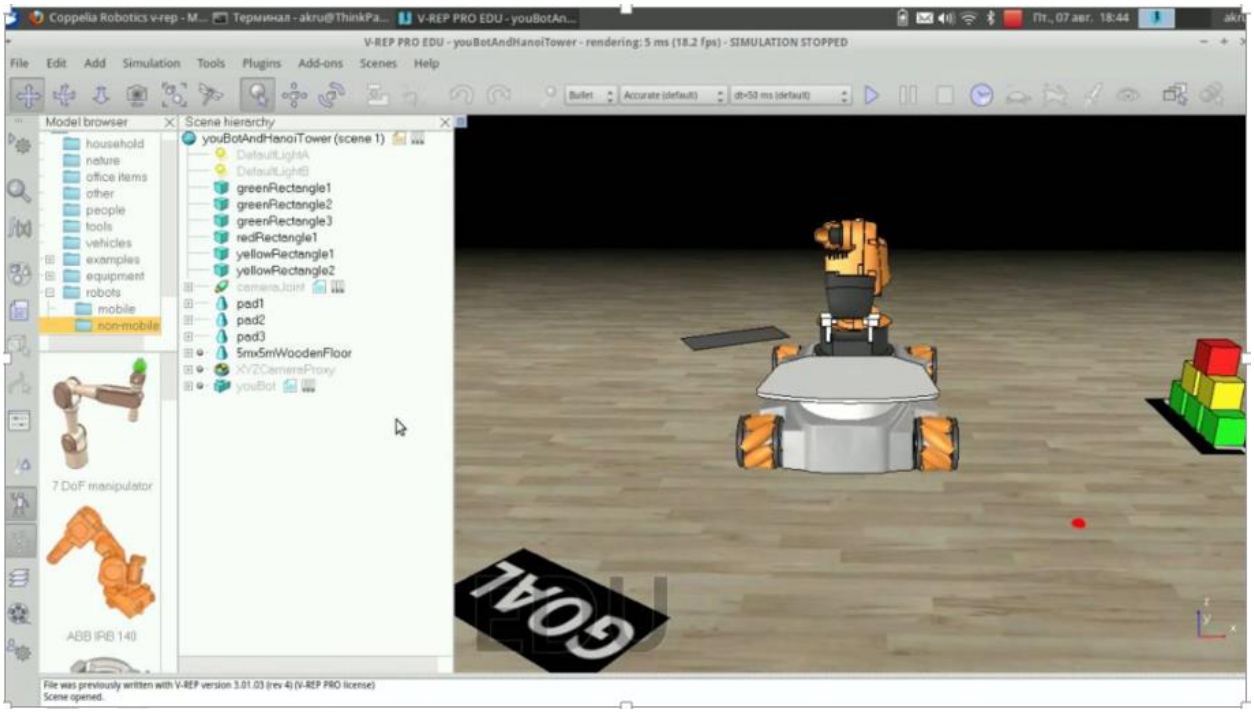


Рисунок 1. Симуляция движения и распознавания цветов в V-REP PRO EDU.

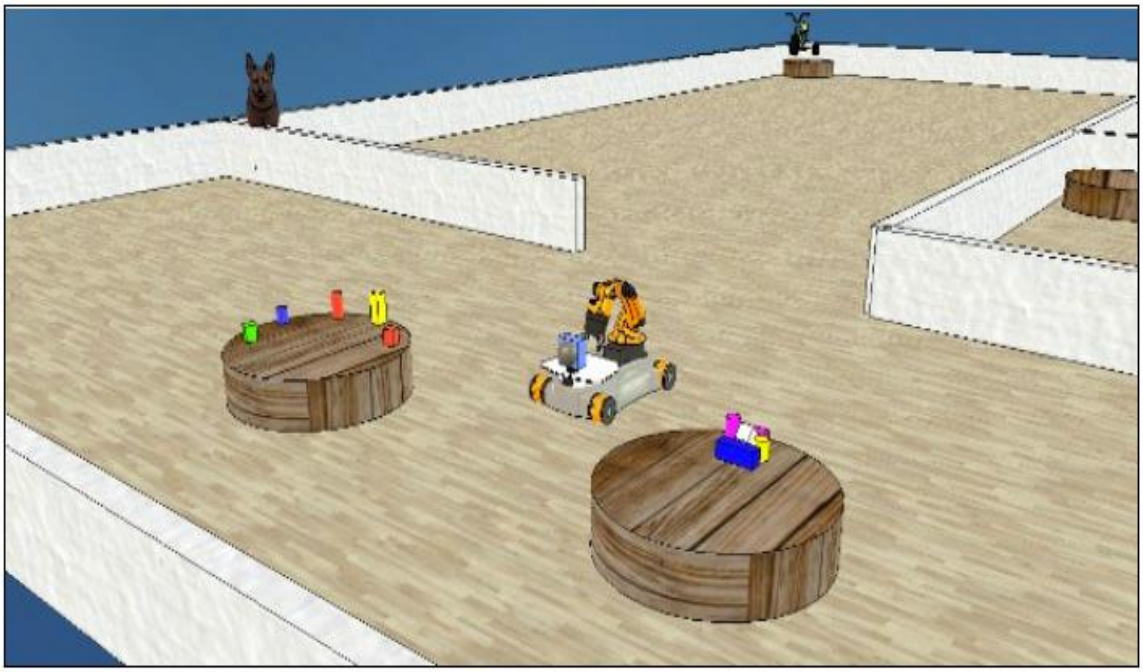


Рисунок 2. Симуляция перемещения робота KUKA youBot в виртуальном доме V-REP PRO EDU.

Задача: KUKA youBot, передвигаясь по виртуальному дому убрать объекты, соответственно месту, где они должны храниться. Объект имеет цилиндрическую или коробчатую форму. Основания ящиков и цилиндров фиксированы, только их высота варьируется. Основание коробчатых объектов составляет 50 мм на 50 мм. Диаметр основания цилиндров составляет 50 мм. Цвет объектов меняется. Сначала объекты размещаются на двух таблицах,

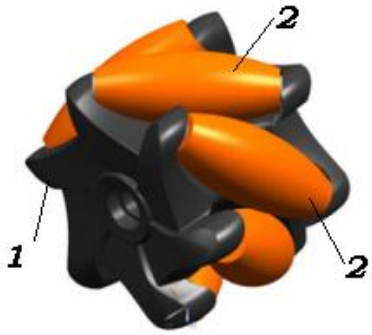
стоящих перед исходной позицией youBot. (Эти две таблицы видны с помощью датчика Hokuyo от начальной позиции youBot. Никакие другие цилиндрические объекты не видны в Hokuyo из исходной позиции youBot.) На одном столе объекты размещаются вертикально. На другом столе объекты не обязательно располагаются вертикально - они могут быть установлены под наклоном.

2. Управление реальным роботом KUKA youBot

Всё чаще мобильные робототехнические системы, предназначенные для работы в стеснённых условиях складских, производственных и подобных помещений, реализуются на базе платформ всенаправленного движения, оснащённых роликонесущими колёсами. Примером такой системы является *KUKA omniMove* – мобильная платформа повышенной грузоподъёмности, оснащённая меканум-колёсами (*Mecanum wheel*). На периферии меканум-колеса располагаются тела качения – ролики, оси которых скрещиваются с осью колеса под углом 45°.

Объектом исследования настоящей работы является *KUKA youBot* (см. Рисунок. 2а) - мобильная платформа всенаправленного движения с меканум-колёсами (см. Рисунок. 2б), оснащённая одним или двумя манипуляторами. Для этого робота создано свободное и открытое программное обеспечение, что позволяет использовать *youBot* для широкого класса научно-исследовательских и учебных задач.

а) Общий вид робота с одним манипулятором с устройством захвата



б) Роликонесущее меканум-колесо
(*1* – диск колеса, *2* - ролики)

Рисунок 3. Мобильный робот всенаправленного движения youBoot

Существующий программно-аппаратный комплекс позволяет организовать управление роботом *youBot* на кинематическом и динамическом уровнях. В первом случае на вход системы управления подаются желаемые величины кинематических параметров (проекции вектора скорости центра платформы, её угловая скорость или угловые скорости вращения колёс). Во втором случае на вход системы управления могут подаваться величины желаемых моментов, которые должны развить приводы, или токов в цепях двигателей или параметры широтно-импульсной модуляции.

Эксперименты демонстрируют неудовлетворительную точность кинематического управления на частотах выше 0,5 рад/с. Повысить точность управления и провести его оптимизацию можно, используя алгоритмы динамического уровня, но для их синтеза требуется проработанная математическая модель системы.

Кинематика платформы всенаправленного движения youBot

Исследуем движение мобильного робота *youBot* по горизонтальной плоскости (подстилающей поверхности, полу). Считаем, что манипулятор неподвижен относительно платформы.

Для описания кинематики платформы мобильного робота введём неподвижную систему координат $X'Y'Z'$ с горизонтальной плоскостью $X'Y'$, и связанную с платформой подвижную OXYZ с началом в геометрическом центре платформы (см. Рисунок. 4).

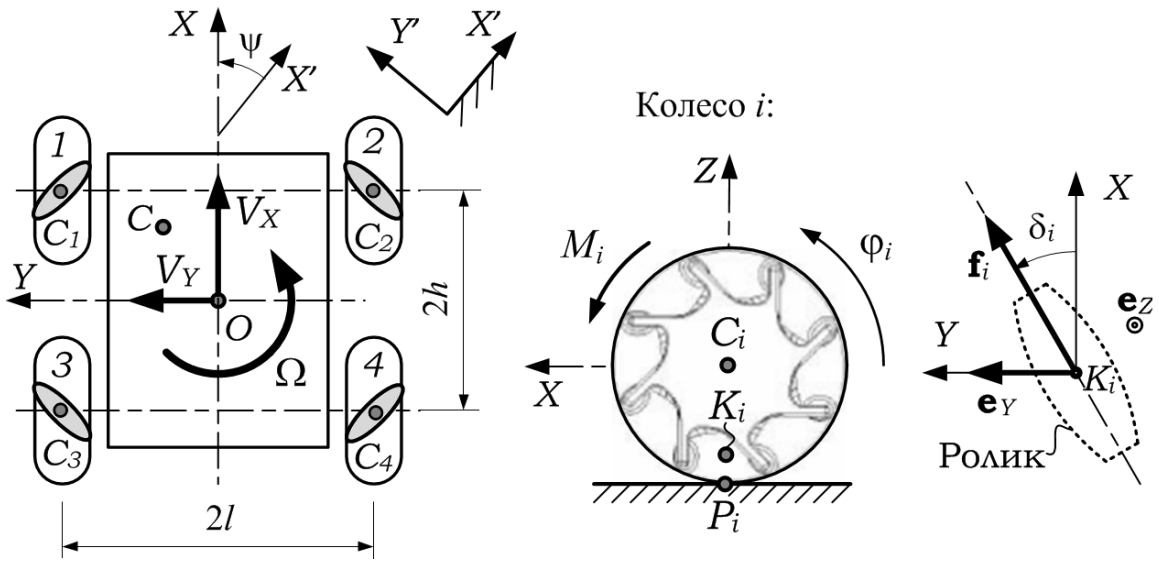


Рисунок 4. Схема мобильного робота (указаны положения роликов, контактирующих с подстилающей поверхностью)

декартовыми

Движение системы описывается обобщёнными координатами координатами X'_O и Y'_O её геометрического центра O; курсового угла $\psi=\angle(X',X)$; углами поворота колёса относительно корпуса робота ϕ_i $(i = 1,...,4)$; углами поворота роликов, контактирующих с подстилающей поверхностью, относительно дисков колёс γ_i $(i = 1,...,4)$.

В рамках настоящей работы рассматривается **идеальная** модель движения меканум-колёс. Считаем, что в каждый момент времени ролик колеса i движется без проскальзывания и отрыва от подстилающей поверхности, причём центр колеса C_i, центр оси ролика K_i и точка контакта P_i ролика с полом лежат на одной вертикальной прямой (см. рис. 2).

В соответствии с принятыми допущениями вектор скорости точки контакта $\vec{V}_{P_i}$ равен [9]:

$$\vec{V}_{P_i} = \vec{V}_O + [\ \vec{\Omega}, \overrightarrow{\mathrm{OC}_i}\ ] + [\ \vec{\omega}_i, \overrightarrow{C_iK_i}\ ] + [\ \vec{\omega}_{p\,i}, \overrightarrow{K_iP_i}\ ] = 0, \tag{1}$$

где $\vec{V}_O = \{V_X, V_Y, 0\}$ - вектор скорости центра платформы; $\vec{\Omega}$, $\vec{\omega}_i$ и $\vec{\omega}_{p\,i}$ – соответственно, векторы абсолютных угловых скоростей платформы, колеса i и его ролика, контактирующего с полом; $[\,,\,]$ - операция векторного умножения. Перечисленные векторы угловых скоростей равны

$$\vec{\Omega} = \Omega\vec{e}_Z, \vec{\omega}_i = \Omega\vec{e}_Z + \dot{\phi}_i\vec{e}_Y, \vec{\omega}_{p\,i} = \Omega\vec{e}_Z + \dot{\phi}_i\vec{e}_Y + \dot{\gamma}_i\vec{f}_i, \tag{2}$$

где $\Omega = \dot{\psi}$ - угловая скорость платформы; $\vec{e}_Y$ и $\vec{e}_Z$ - орты осей координатных X и Y; $\vec{f}_i = \{\cos\delta_i, \sin\delta_i, 0\}$ - единичный вектор оси ролика колеса i (см. рис. 2), $\delta_2 = \delta_3 = 45°$, $\delta_1 = \delta_4 = -45°$. Здесь координаты векторов указаны в подвижной системе OXYZ.

Выполнив необходимые преобразования для каждого из колёс $(i = 1,...,4)$, из формул (1) и (2) получаем выражения для угловых скоростей вращения колёс и роликов:

$$\dot{\phi}_1 = \frac{1}{R}[V_X - V_Y - (l+h)\Omega], \quad \dot{\phi}_2 = \frac{1}{R}[V_X + V_Y + (l+h)\Omega], \quad \dot{\gamma}_1 = \dot{\gamma}_2 = -\frac{\sqrt{2}}{r}[V_Y + h\,\Omega$$
$$\dot{\phi}_3 = \frac{1}{R}[V_X + V_Y - (l+h)\Omega], \quad \dot{\phi}_4 = \frac{1}{R}[V_X - V_Y + (l+h)\Omega], \quad \dot{\gamma}_3 = \dot{\gamma}_4 = -\frac{\sqrt{2}}{r}[V_Y - h\,\Omega$$
(3)

где $\boldsymbol{R} = |\mathrm{C}_i\boldsymbol{P}_i|$ - радиус меканум-колеса, $\boldsymbol{r} = |\boldsymbol{K}_i\boldsymbol{P}_i|$ - радиус поперечного сечения ролика, $\boldsymbol{l}$ и $\boldsymbol{h}$ - характерные размеры платформы (см. рис. 2).

Соотношения (3) обосновывают всенаправленное свойство движения платформы - произвольные законы изменения $\boldsymbol{V_X}$, $\boldsymbol{V_Y}$ и $\boldsymbol{\Omega}$ могут быть реализованы за счёт вращения роликонесущих колёс. Например, для поступательного движения робота вбок ($\boldsymbol{V_X} = \boldsymbol{0}$, $\boldsymbol{\Omega} = \boldsymbol{0}$) вращение колёс

должно производиться со скоростями $\dot{\boldsymbol{\phi}}_{\boldsymbol{2}} = \dot{\boldsymbol{\phi}}_{\boldsymbol{3}} = -\dot{\boldsymbol{\phi}}_{\boldsymbol{1}} = -\dot{\boldsymbol{\phi}}_{\boldsymbol{4}} = \boldsymbol{V_Y/R}$, а для вращения вокруг центра платформы ($\boldsymbol{V_X = 0}$, $\boldsymbol{V_Y = 0}$) – со скоростями $\dot{\boldsymbol{\phi}}_{\boldsymbol{2}} = \dot{\boldsymbol{\phi}}_{\boldsymbol{4}} = -\dot{\boldsymbol{\phi}}_{\boldsymbol{1}} = -\dot{\boldsymbol{\phi}}_{\boldsymbol{3}} = = (\boldsymbol{l + h})\,\boldsymbol{\Omega/R}$.

Модель динамики робота youBot

Составим уравнения движения робота, считая, что в сочленениях тел действуют линейные по скорости силы вязкого трения, а ролики и вращающиеся детали механических передач колёс невесомы. Каждое колесо рассматриваемой системы оснащено отдельным электроприводом. Моменты, развиваемые этими приводами обозначим $\boldsymbol{M_i}$ $(\boldsymbol{i = 1,\ldots,4})$ (см. рис. 2).

Соотношения (3) определяют восемь неголономных связей, налагаемых на динамику мобильного робота *youBot*. Из его 11 обобщённых скоростей можно составить лишь три независимые линейные комбинации, например,

$$\boldsymbol{V_X \equiv \dot{X}'_O \cos\psi + \dot{Y}'_O \sin\psi, V_Y \equiv -\dot{X}'_O \sin\psi + \dot{Y}'_O \cos\psi, \Omega \equiv \dot{\psi},}$$

и с помощью них описать динамику системы.

Для составления уравнений движения робота воспользуемся уравнениями Аппеля [10; 11].

Построение уравнений Аппеля для исследуемой системы подробно изложено в работе [12].

С учётом введённых выше допущений, динамика мобильного робота *youBot* описывается следующей системой уравнений:

$$m_0(\dot{V}_X - V_Y\Omega) + \frac{4I_1\dot{V}_X}{R^2} - \mathrm{ma}_Y\dot{\Omega} - \mathrm{ma}_X\Omega^2 + \frac{4\mu_1 V_X}{R^2} = F_X, \quad (4а)$$

$$m_0(\dot{V}_Y + V_X\Omega) + \frac{4I_1\dot{V}_Y}{R^2} + \mathrm{ma}_X\dot{\Omega} - \mathrm{ma}_Y\Omega^2 + \left(\frac{4\mu_1}{R^2} + \frac{8\mu_2}{r^2}\right)V_Y = F_Y, \quad (4б)$$

$$\left(I_0 + \frac{4I_1(l+h)^2}{R^2}\right)\dot{\Omega} - \mathrm{ma}_Y(\dot{V}_X - V_Y\Omega) + \mathrm{ma}_X(\dot{V}_Y + V_X\Omega) + \left(\frac{4\mu_1(l+h)^2}{R^2} + \frac{8\mu_2 h^2}{r^2}\right)\Omega = M_\Omega, \quad (4в)$$

где $m_0 = m + 4m_1$ – общая масса робота; I_0 – его момент инерции относительно оси OZ; m – масса платформы со всем перевозимым оборудованием и грузом; a_X и a_Y – координаты её центра масс C в подвижной системе OXY (рис. 2); I_1 – момент инерции колеса относительно его оси; m_1 - масса колеса; μ_1 – коэффициент вязкого трения в сочленениях колёс и платформы; μ_2 - коэффициент вязкого трения в осях роликов; F_X, F_Y и

M_Ω - управляющие обобщённые силы, которые выражаются через моменты приводов колёс M_i, $i = 1...4$, по формулам

$$F_X = \frac{M_2 + M_1 + M_4 + M_3}{R}, F_Y = \frac{M_2 - M_1 - M_4 + M_3}{R}, M_\Omega = \eta_1[M_2 - M_1 + M_4 - M_3].$$

Значения некоторых коэффициентов в системе (4) неизвестны и подлежат оцениванию.

Для удобства дальнейшего анализа, введём новые переменные:

$$V_X = (l + h)\Omega_{V_X}, V_Y = (l + h)\Omega_{V_Y}, M_{V_X} = (l + h)F_X, M_{V_Y} = (l + h)F_Y,$$

и перепишем систему уравнений движения (4) в следующей форме:

$$y = H\,x, \tag{5}$$

где $x = (x_i) = \left(m_0(l+h)^2 \quad I_0 \quad \frac{4I_1(l+h)^2}{R^2} \quad \mathrm{ma}_X(l+h) \quad \mathrm{ma}_Y(l+h) \quad \frac{4\mu_1(l+h)^2}{R^2} \quad \frac{8\mu_2 h^2}{r^2}\right)^T$ – вектор неизвестных постоянных параметров (коэффициентов в уравнениях движения);

$$y = \begin{pmatrix} M_{V_X} \\ M_{V_Y} \\ M_\Omega \end{pmatrix}, H = \begin{bmatrix} \dot{\Omega}_{V_X} - \Omega_{V_Y}\Omega & 0 & \dot{\Omega}_{V_X} & -\Omega^2 & -\dot{\Omega} & \Omega_{V_X} & 0 \\ \dot{\Omega}_{V_Y} + \Omega_{V_X}\Omega & 0 & \dot{\Omega}_{V_Y} & \dot{\Omega} & -\Omega^2 & \Omega_{V_Y} & \eta^2\Omega_{V_Y} \\ 0 & \dot{\Omega} & \dot{\Omega} & \Omega_{V_X}\Omega + \dot{\Omega}_{V_Y} & \Omega_{V_Y}\Omega - \dot{\Omega}_{V_X} & \Omega & \Omega \end{bmatrix};$$

$\eta = (l + h)/h$ – безразмерная длина.

ЗАДАНИЕ ДЛЯ РАЗДЕЛА

«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8ЕМ61	Ефименко Анастасии Павловне

Школа	ИШИТР	**Отделение**	Автоматизации и робототехники
Уровень образования	Магистр	**Направление/специальность**	Мехатроника и робототехника

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	*Работа с информацией, представленной в российских и иностранных научных публикациях, аналитических материалах, статистических бюллетенях и изданиях, нормативно-правовых документах; анкетирование; опрос*
2. *Нормы и нормативы расходования ресурсов*	
3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого потенциала, перспективности и альтернатив проведения НИ с позиции ресурсоэффективности и ресурсосбережения*	*4. Определение трудоёмкости выполняемых работ*
2. *Проведение анализа НИ с помощью метода стратегического планирования*	*5. Определение ресурсной, финансовой и экономической эффективности исследования*
3. *Планирование и формирование бюджета научных исследований*	
Перечень графического материала *(с точным указанием обязательных чертежей):*	
1. *Исследование целесообразности вложения денежных средств в НИ.* 2. *Матрица SWOT.* 3. *Альтернативы проведения НИ* 4. *График проведения (сетевой и календарный план-графики) и бюджет НИ* 5. *Оценка ресурсной, финансовой и экономической эффективности НИ*	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент отделения СГН	Петухов О.Н.	к.э.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ЕМ61	Ефименко Анастасия Павловна		

3 Оценка коммерческого потенциала и перспективности проведения научных исследований с позиции ресурсоэффективности и ресурсосбережения

3.1.1. Потенциальные потребители результатов исследования

Научные исследования, проведенные в ходе выполнения ВКР, направлены на получение опыта студентами младших курсов для последующих исследований и на использование на предприятиях, в производственных процессах которых целесообразно применение мобильных роботов, способных передвигаться и распознавать объекты.

Конечным потребителем результатов исследований является студент, занимающийся исследованием мобильных роботов и проектированием систем управления ими. Также потребителем может быть проектировщик систем автоматического управления на предприятии.

3.1.2. Исследование целесообразности вложения денежных средств в научно-исследовательский проект

Воспользуемся технологией QuaD (Quality Advisor) для определения целесообразности вложения денег в реализацию результатов проведенных исследований. Для этого, выделим ряд наиболее ярко характеризующих проекты такого рода показатели и определим их вклад в общую оценку перспективности проекта (весовой коэффициент). Весовые коэффициенты для каждого критерия будем определять исходя из личного опыта.

Во внимание были приняты следующие показатели: надёжность построенной модели (0,15); безопасность (0,15); простота эксплуатации (0,1); адекватность результатов, получаемых при применении предложенной методики (0,15); качество программного обеспечения, реализующего предложенную методику (0,17); простота внедрения методики и соответствующего ПО на предприятии (0,08); актуальность проведенных исследований для предприятия-потребителя (0,1); минимально необходимая для работы по разработанной методике квалификация персонала (0,08); наличие альтернативных способов применения разработки (0,05); конкуренция со стороны прочих разработчиков (0,02).

Результаты проведённого сравнения выбранных показателей представлены в таблице 3.

Таблица 3 – Оценка перспективности разработки

Критерий оценки	Вес критерия	Баллы	Максимальный балл	Относительное значение	Средневзвешенное значение
1. Надёжность	0,15	80	100	0,8	12
2. Безопасность	0,15	80	100	0,8	12
3. Простота эксплуатации	0,1	80	100	0,8	8
4. Адекватность результатов	0,15	100	100	1	15
5. Качество ПО	0,15	100	100	1	15
6. Простота внедрения	0,05	80	100	0,8	4
7. Функциональная мощность	0,15	90	100	0,9	13,5
8. Потребность в ресурсах памяти	0,04	80	100	0,8	3,2
9. Энергоэкономичность	0,04	80	100	0,8	3,2
10. Отсутствие конкурентов	0,02	30	100	0,3	0,6
Итого	1	800	1000	8	86,5

Оценка качества и перспективности по технологии QuaD определяется по формуле:

$П_{ср} = \sum B_i \times Б_i$,

где $П_{ср}$ – средневзвешенное значение показателя качества и перспективности научной разработки; B_i – вес показателя (в долях единицы); $Б_i$ – средневзвешенное значение i-го показателя. Значение $П_{ср}$ позволяет говорить о перспективах разработки и качестве проведенного исследования. Если значение показателя $П_{ср}$ получилось от 100 до 80, то такая разработка считается перспективной. Если от 79 до 60 – то перспективность выше среднего. Если от 59 до 40 – то перспективность средняя. Если от 39 до 20 – то перспективность ниже среднего. Если 19 и ниже – то перспективность крайне низкая.

Таким образом, по результатам применения технологии QuaD, созданную в рамках выполнения ВКР методику и соответствующее ПО можно считать перспективной разработкой, так как значение средневзвешенной оценки оказалось больше восьмидесяти.

3.1.3. SWOT-анализ

SWOT-анализ заключается в выявлении сильных и слабых сторон проекта, возможностей для дальнейшего развития и угроз существованию; направлен на исследование внутренней и внешней среды проекта.

Исследование проводится в три этапа: определение сильных и слабых сторон, угроз и возможностей проекта; построение и анализ интерактивных матрицы проекта; построение итоговой матрицы SWOT-анализа.

Таблица 4 – SWOT-анализ

	С1 Универсальность робота KUKA youBot С2 Возможность усложнения конструкции робота введением дополнительных датчиков	Сл1 Отсутствие информации об аналогичных разработках Сл2 Отсутствие базовых знаний для работы с KUKA youBot
В1 Обмен опытом с другими разработчиками В2 Усложнение конструкции робота введением дополнительных датчиков	Распространённость робота KUKA youBot позволяет разработчикам обмениваться опытом. Возможность Усложнение конструкции робота введением дополнительных датчиков позволит повысить сложность выполняемых задач	Отсутствие базовых знаний для работы с KUKA youBot
У1 Быстрое освоение новых задач другими разработчиками У2 Прекращение функционирования форума разработчиков KUKA youBot	Прекращение функционирования форума разработчиков KUKA youBot приведёт к отсутствию обмена данными и к совместной деятельности, а также к прекращению сервисной поддержки производителей Быстрое освоение новых задач другими разработчиками	Изменение конструкции исследуемых технических средств может повлечь отказ целевого предприятия от использования разработки, однако, общая закрытость отрасли и отсутствие выхода на зарубежные рынки не позволит продавать разработку в том же виде, следовательно, необходимо сделать предлагаемую разработку наиболее универсальной.

		Отсутствие на предприятии сотрудника, способного работать с результатами исследований, и отсутствие возможности поддерживать оборудование заказчика в течение его эксплуатации может привести к отсутствию спроса на разработку.

3.2. Определение возможных альтернатив проведения исследований

Воспользуемся морфологическим подходом для определения возможных альтернатив проведения исследования.

Упомянутый подход предполагает точную формулировку проблемы исследования, раскрытие важных морфологических характеристик объекта исследования и раскрытие вариантов по каждой характеристике.

Проблема исследования, проводимого в рамках ВКР, заключается в решении задачи синтеза параметров регулятора автоматической системы компенсации веса. Данная задача может быть решена в несколько этапов:

– управление KUKA youBot;

– регулятор;

– язык программирования;

–среда программирования;

– проверка результатов.

Проанализируем возможные варианты прохождения через эту последовательность шагов. Полученные результаты сведем в таблицу.

Таблица 5 – Альтернативные варианты проведения исследований

	1	2
А. Управление KUKA youBot	Реальный робот	Симулятор

Б. Разработка ПО	Ноутбук	ПК
В. Регулятор	ПИ-регулятор	ПИД-регулятор
Г. Язык программирования	C++	Lua
Д. Среда программирования	Eclipse	V-REP
Е. Проверка результатов	Проверка на роботе	Симуляция

Среди всех возможных согласно таблице 3 вариантов выберем два наиболее рациональных варианта проведения исследований:

– А1Б1В2Г1Д1Е1 – точное моделирование; на ноутбуке процесс будет выполняться без привязки к рабочему месту

выбор уже изученного корневого подхода к синтезу позволяет сразу приступить к дальнейшим исследованиям без изучения дополнительного теоретического материала; ПИ-регулятор обеспечивает переходные процессы без перерегулирования; наличие мощного аппарата математических операций в языке Matlab упрощает разработку ПО; отсутствие необходимости проводить эксперимент на реальном объекте усложняет проверку полученных результатов, однако, удешевляет разработку;

– А2Б2В1Г2Д2Е2 –

экспериментальное определение модели объекта даст более простую, но менее точную модель объекта; выбор частотного подхода к синтезу приведет к необходимости изучения дополнительного материала и усложнит программную реализацию метода синтеза; ПИД-регулятор обеспечит максимально быстрые переходные процессы; программное обеспечение разработанное на С в Visual Studio, в отличие от разработанного в Matlab, будет полностью автономно, однако, это усложнит отладку программы и резко увеличит сроки ее разработки; отсутствие необходимости проводить эксперимент на реальном объекте усложняет проверку полученных результатов, однако, удешевляет разработку.

3.3. Планирование научно-исследовательских работ

3.3.1 Структура работ в рамках научного исследования

Планирование комплекса предполагаемых работ осуществляется в следующем порядке:

– определение структуры работ в рамках научного исследования;

– определение участников каждой работы;

– установление продолжительности работ;

– построение графика научных исследований.

Рабочая группа, выполняющая научные исследования, состоит из двух человек: научного руководителя, непосредственного исполнителя – инженера.

Таблица 6 – Перечень работ и распределение исполнителей

Основные этапы	№ раб	Содержание работ	Должность исполнителя
Разработка технического задания	1	Составление и утверждение технического задания	Руководитель, инженер
Выбор направления исследований	2	Подбор и изучение материалов по теме	Инженер
	3	Выбор направления исследований	Руководитель, инженер
Теоретические исследования	4	Изучение научной литературы по выбранному направлению	Инженер
	5	Изучение документации на робота KUKA youBot	Инженер
	6	Решение задачи стабилизации скорости и стабилизации положения робота	Инженер
	7	Разработка ПО для управления роботом	Инженер
	8	Тестирование разработанного ПО и устранение ошибок	Инженер
Оценка результатов исследований	9	Оценка результатов исследований	Инженер
	10	Определение целесообразности проведения исследований	Руководитель, инженер
Оформление результатов исследований	11	Составление отчета по НИР	Инженер
	12	Государственная регистрация ПО	Инженер

3.3.2. Определение трудоемкости выполнения работ

Трудовые затраты в большинстве случаях образуют основную часть стоимости разработки, поэтому важным моментом является определение трудоемкости работ каждого из участников научного исследования.

Трудоемкость выполнения научного исследования оценивается экспертным путем в человеко-днях и носит вероятностный характер, т.к. зависит от множества трудно

учитываемых факторов. Для определения ожидаемого (среднего) значения трудоемкости $t_{ожi}$ используется следующая формула:

$$t_{ожі} = \frac{3t_{minі}+2t_{maxі}}{5},$$

где $t_{ожі}$ – ожидаемая трудоемкость выполнения *i*-ой работы, чел.-дн.;

$t_{minі}$ – минимально возможная трудоемкость выполнения заданной *i*-ой работы (оптимистическая оценка: в предположении наиболее благоприятного стечения обстоятельств), чел.-дн.;

$t_{maxі}$ – максимально возможная трудоемкость выполнения заданной *i*-ой работы (пессимистическая оценка: в предположении наиболее неблагоприятного стечения обстоятельств), чел.-дн.

Исходя из ожидаемой трудоемкости работ, определяется продолжительность каждой работы в рабочих днях $T_р$, учитывающая параллельность выполнения работ несколькими исполнителями:

$$T_{рi} = \frac{t_{ожі}}{Ч_i},$$

где $T_{рi}$ – продолжительность одной работы, раб. дн.;

$t_{ожі}$ – ожидаемая трудоемкость выполнения одной работы, чел.-дн.

$Ч_i$ – численность исполнителей, выполняющих одновременно одну и ту же работу на данном этапе, чел.

длительность каждого из этапов работ из рабочих дней следует перевести в календарные дни. Для этого необходимо воспользоваться следующей формулой:

$$T_{кі} = Т_{рі} \cdot k_{кал},$$

где $T_{кі}$– продолжительность выполнения *i*-й работы в календарных днях;
$T_{рі}$ – продолжительность выполнения *i*-й работы в рабочих днях;
$k_{кал}$ $k_{кал}$– коэффициент календарности.
Коэффициент календарности определяется по следующей формуле:

$$k_{кал} = \frac{T_{кал}}{Т_{кал}-Т_{вых}-Т_{пр}},$$

где $T_{кал}$ – количество календарных дней в году;

$Т_{вых}$ – количество выходных дней в году;

$Т_{пр}$ – количество праздничных дней в году.

В 2014 году 365 дней; из них 119 выходных и праздничных дней дня; 14 праздничных дней. Коэффициент календарности рассчитаем следующим образом:

$$k_{кал} = \frac{365}{365\text{-}119} = 1{,}48.$$

Рассчитанные значения в календарных днях по каждой работе $T_{кі}$ необходимо округлить до целого числа.

Результаты расчетов сведем в таблицу.

Таблица 7 – Расчет трудоемкости выполняемых работ

Название работы	Трудоёмкость работ						Исполнители	Длительность работ в рабочих днях T_{pi}		Длительность работ в календарных днях T_{ki}	
	t_{min}, чел-дни		t_{max}, чел-дни		$t_{ожi}$, чел-дни						
	НР	И	НР	И	НР	И	НР – научный руководитель; И – инженер	НР	И	НР	И
1 Составление и утверждение технического задания	1	2	2	4	1,4	2,8	НР, И	1,4	2,8	2	4
2 Подбор и изучение материалов по теме	-	10	-	15	-	12	И	-	12	-	18
3 Выбор направления исследований	2	3	3	5	2,4	3,2	НР, И	2,4	3,2	4	5
4 Освоение ОС Linux и симулятора V-REP	-	10	-	16	-	12,4	И	-	12,4	-	18
5 Написание программы в симуляторе	-	6	-	11	-	8	И	-	8	-	9
6 Написание программы для распознавания объектов	1	7	3	10	2,4	8,2	И	2,4	8,2	4	12
7 Написание программы для движения по заданной траектории	-	9	-	12	-	10,2	И	-	10,2	-	15

8 Отладка разработанного ПО	-	2	-	4	-	2,8	И	-	2,8	-	4
9 Оценка результатов	-	1	-	2	-	1,4	И	-	1,4	-	2
10 Составление отчета по НИР	-	2	-	5	-	5,2	И	-	3,2	-	5
11 Презентация работы	-	1	-	2	-	1,4	И	-	1,4	-	2
Итого								6,2	65,6	10	94

3.3.3. Составление календарного план-графика работ

На основании расчетов трудоемкости проводимых исследований в различных исполнениях построим календарный план-график наиболее трудоемкого исполнения. Прямоугольниками со штриховкой будем обозначать трудозатраты инженера, прямоугольниками без штриховки – трудозатраты научного руководителя.

Таблица 8 – Календарный план-график работ

№	Вид работ	Исполнители	T_{ki}, кал. дн.	Продолжительность выполнения работ												
				февр.		март			апрель			май			июнь	
				2	3	1	2	3	1	2	3	1	2	3	1	2
1	Составление и утверждение технического задания	НР	2													
		И	4													
2	Подбор и изучение материалов по теме	И	23													
3	Выбор направления исследований	НР	4													
		И	5													
4	Освоение ОС Linux и симулятора V-REP	И	23													
5	Написание программы в симуляторе	И	9													
6	Написание программы для распознавания объектов	НР	4													
		И	15													
7	Написание программы для движения по заданной траектории	И	23													
8	Отладка разработанного	И	2													

	ПО															
9	Оценка результатов	И	2													
10	Составление отчета по НИР	И	5													
11	Презентация работы	И	2													

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8ЕМ61	Ефименко Анастасии Павловне

Школа	ИШИТР	**Отделение**	Автоматизации и робототехники
Уровень образования	Магистр	**Направление/специальность**	Мехатроника и робототехника

Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения	Объектом исследования является мобильная роботизированная платформа KUKA youBot на всенаправленных колёсах. Данного робота в дальнейшем предполагается применять в различных исследованиях (изучение и разработка алгоритмов управления) и сферах производства (автомобильная, космическая, атомная промышленность и т.д.) Конечным продуктом является программный обеспечение для управления роботом.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Производственная безопасность 1.1. Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения	Анализ выявленных вредных факторов: • повышенный уровень шума; • повышенный уровень электромагнитных излучений; • повышенная или пониженная влажность воздуха; • недостаточная освещённость рабочей зоны; • отсутствие естественного света.
1.2. Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения	Анализ выявленных опасных факторов: • Движущиеся машины и механизмы; подвижные части производственного оборудования; • электрический ток (источником является ПК);
2. Экологическая безопасность: –	Воздействие объекта на атмосферу, гидросферу не происходит. В работе проведён анализ воздействия на литосферу (образование отходов при поломке компонентов ПК).
3. Безопасность в чрезвычайных ситуациях: –	В офисном помещении (кабинете) возможно ЧС техногенного характера – пожар (возгорание).
4. Правовые и организационные вопросы обеспечения безопасности: –	Рабочее место при выполнении работ в положении сидя должно соответствовать требованиям ГОСТ 12.2.032-78. Требования к организации оборудования рабочих мест с ПК регулируется в СанПиН 2.2.2/2.4.1340 – 03.

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОКД ИШНКБ	Бородин Ю.В.	к.т.н., доцент		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ЕМ61	Ефименко Анастасия Павловна		.

Введение

В данном разделе рассматриваются вопросы обеспечения комфортных и безопасных условий труда специалиста, осуществляющего разработку программного обеспечения для KUKA youBot. Конечным продуктом является программное обеспечение контроллера робота для управления роботом KUKA youBot. Решение предполагается применять для обучения студентов и в различных сферах промышленности, где необходимо применять роботов, которые должны перемещаться, осуществлять действия с различными объектами, обладать техническим зрением. Исследования и разработка программного обеспечения осуществляются в закрытом, отапливаемом и вентилируемом помещении, на рабочем месте, оснащенном персональным компьютером.

Далее будут рассмотрены факторы производственной среды и организация рабочего места, влияющие на работоспособность сотрудника, а также влияние проектной деятельности на состояние окружающей среды и мероприятия, обеспечивающие безопасность в чрезвычайных ситуациях.

4.1. Производственная безопасность

4.1.1. Анализ выявленных вредных факторов при разработке и эксплуатации проектируемого решения:

4.1.1.1. Повышенный уровень шума

Повышенный шум в пределах 40-70 дБ создаёт значительную нагрузку на нервную систему, вызывая ухудшение самочувствия и при длительном воздействии может стать причиной неврозов. Воздействие шума с уровнем шума свыше 80 дБ может привести к потере слуха. При воздействии высоких уровней более 140 дБ возможен разрыв барабанных перепонок, контузия или смерть. Шумовое загрязнение среды на рабочем месте неблагоприятно воздействует на работающих: снижается внимание, снижается скорость психических реакций и т. п. В результате снижается производительность труда и качество выполняемой работы.

При выполнении работ, описанных выше, специалист может оказаться под шумовым воздействием со стороны оборудования, находящегося в рабочем помещении: персональные компьютеры, электрические машины и др.

Работы, выполняемые специалистом, оцениваются как научная деятельность, конструирование и проектирование. В связи с этим эквивалентный уровень шума в рабочем помещении не должен превышать 50дБА [1].

Наиболее эффективная защита от производственного шума создается с помощью специальных архитектурно-строительных решений на этапе проектирования здания, планировки офиса. В качестве дополнительных мер по защите от шума можно применят различные звукоизолирующие кожухи, звукопоглощающие отделочные материалы.

4.1.1.2. Повышенный уровень электромагнитных излучений; повышенная напряжённость электрического поля

При длительном действии электромагнитного поля различных диапазонов длин волн характерно развитие функциональных расстройств центральной нервной системы, неярко выраженные сдвиги эндокринно-обменных процессов, изменение состава крови, трофические нарушения и стойкое снижение работоспособности.

Источником электромагнитного поля и электромагнитных излучений на рабочем месте является компьютер, в частности экран монитора компьютера, процессор и клавиатура. Вокруг ПК образуется электромагнитное поле с диапазоном частот от 5 до 400 кГц/ Мощность экспозиционной дозы мягкого рентгеновского излучения в любой точке на расстоянии 0,05 м от экрана при любых положениях ПК не должна превышать 100 мкР/час [2]. Допустимые значения параметров неионизирующих электромагнитных излучений от монитора компьютера представлены в таблице 9.

Таблица 9 – Допустимые значения параметров неионизирующих электромагнитных излучений [2]

Наименование параметра	Допустимые значения
Напряженность электрической составляющей электромагнитного поля на расстоянии 50см от поверхности видеомонитора	10 В/м
Напряженность магнитной составляющей электромагнитного поля на расстоянии 50см от поверхности видеомонитора	0,3 А/м
Напряженность электростатического поля не должна превышать: – для взрослых пользователей	20 кВ/м

Предельно-допустимые нормы ЭМП, создаваемых ПЭВМ, представлены в таблице 2.

Таблица 10 – Предельно допустимые нормы ЭМП, создаваемых ПЭВМ [2]

Напряжённость электрического поля	
в диапазоне частот 5 Гц−2 кГц	25 В/м
в диапазоне частот 2 кГц−400 кГц	2,5 В/м
Плотность магнитного потока	
в диапазоне частот 5 Гц−2 кГц	250 нТл
в диапазоне частот 2 кГц−400 кГц	25 нТл

Ряд мероприятий, позволяющих уменьшить влияние вредных факторов на работника при работе за ПК: каждые 45–60 минут необходимо делать перерыв на 10–15, выполнять гимнастику для глаз, а также выполнять упражнения на расслабление, которые могут уменьшить напряжение, накапливающееся в мышцах при длительной работе за компьютером.

4.1.1.3. Повышенная или пониженная влажность воздуха

Повышенная влажность при высокой температуре воздуха способствует перегреванию организма, при низкой температуре увеличивается теплоотдача с поверхности кожи, что ведет к переохлаждению. Низкая влажность вызывает неприятные ощущения в виде сухости слизистых оболочек дыхательных путей работающего.

Выполняемые работы по интенсивности энергозатрат попадают в категорию Iа, так как выполняются сидя и без значительных физических напряжений. Таким образом, оптимальными нужно считать параметры микроклимата, соответствующие категории Iа в таблице 3 и 4.

Таблица 3 - Оптимальные величины показателей микроклимата на рабочих местах производственных помещений [3]

Период года	Категория работ по уровню энергозатрат, Вт	Температура воздуха, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Холодный	Iа (до 139)	22-24	40-60	0,1
Тёплый	Iа (до 139)	23-25		0,1

Таблица 11 - Допустимые величины показателей микроклимата на рабочих местах производственных помещений [3]

Период года	Категория работ по уровню энергозатрат, Вт	Температура воздуха, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Холодный	Iа (до 139)	20,0 - 21,9	15 - 75	0,1
Тёплый	Iа (до 139)	21,0 - 22,9		0,1

Для поддержания оптимальных параметров микроклимата необходимо применять системы отопления, вентиляции и кондиционирования, увлажнители воздуха, в рабочих помещениях с ПЭВМ необходимо ежедневно проводить влажную уборку и каждый час проветривать помещение.

4.1.1.4. Недостаточная освещённость рабочей зоны; отсутствие или недостаток естественного света

Неудовлетворительное освещение является одной из причин повышенного утомления, особенно при напряжённых зрительных работах. Недостаточное освещение также влияет на психику человека, вызывает усталость центральной нервной системы.

Освещение рабочего места складывается из естественного и искусственного освещения. Естественное освещение достигается установкой оконных проемов с коэффициентом естественного освещения КЕО не ниже 1,2% в зонах с устойчивым снежным покровом и не ниже 1,5% на остальной территории. Световой поток из оконного проема должен падать на рабочее место оператора с левой стороны.

Работа за ПК относится к зрительным работам высокой точности для любого типа помещений [4]. Параметры освещённости:

наименьший размер объекта различения – 0,3 – 0,5 мм;

разряд зрительной работы – III В;

освещенность при системе комбинированного искусственного освещения на рабочей поверхности 600 – 750 лк, в том числе от общего – 200 лк;

коэффициент пульсации освещённости – 5%;

КЕО при верхнем освещении – 2,5 %;

КЕО при боковом освещении – 0,7 %.

Искусственное освещение в помещениях эксплуатации компьютеров должно осуществляться системой общего равномерного освещения.

Требования к освещению на рабочих местах, оборудованных ПК представлены в таблице 12.

Таблица 12 – Требования к освещению на рабочих местах, оборудованных ПК [2]

Освещенность на рабочем столе	300-500 лк
Освещенность на экране ПК	не выше 300 лк

Блики на экране	не выше 40 кд/м^2
Прямая блесткость источника света	не выше 200 кд/м^2
Показатель ослеплённости	не более 20
Показатель дискомфорта	не более 15
Отношение яркости:	
– между рабочими поверхностями	3:1–5:1
– между поверхностями стен и оборудования	10:1
Коэффициент пульсации:	не более 5%

4.2.1. Анализ выявленных опасных факторов при разработке и эксплуатации проектируемого решения:

4.2.1.1. Электрический ток

Категория помещения по опасности поражения электрическим током по ПУЭ, изд.7: помещения без повышенной опасности**,** в которых отсутствуют условия, создающие повышенную или особую опасность;

Электробезопасность – система организационных и технических мероприятий, а также средств, обеспечивающих защиту людей от вредного и опасного воздействия электрического тока, электрической дуги, электромагнитного поля и статистического электричества.

Токи статического электричества, наведенные в процессе работы компьютера на корпусах монитора, системного блока, клавиатуры могут приводить к разрядам при прикосновении к этим элементам. Такие разряды представляют опасность для ПК и могут привести к его выходу из строя.

На рабочем месте пользователя размещены дисплей, клавиатура и системный блок. Перед началом работы следует убедиться в отсутствии свешивающихся со стола или висящих под столом проводов электропитания, в целостности вилки и провода электропитания, в отсутствии видимых повреждений аппаратуры и рабочей мебели, в отсутствии повреждений и наличии заземления при экранного фильтра.

К методам защиты от воздействия статического электричества относятся: влажная уборка, предназначенная для уменьшения количества пылинок в воздухе и на предметах помещения; увеличение влажности воздуха до 65–75 %; защитное заземление электропроводных элементов оборудования; применение средств индивидуальной защиты, таких как антистатические спреи и браслеты.

Допустимый ток частотой 50 Гц при длительности воздействия более 10 секунд составляет 2 мА, а при длительности 10 секунд и менее – 6 мА. Для постоянного тока эта величина равна 10 и 15 мА.

К методам защиты от опасности поражения электрическим током относятся: электрическая изоляция токоведущих частей (сопротивление изоляции должно быть не менее 0,5 МОм для сетей до 1 кВ); ограждение токоведущих частей, которые работают под напряжением; использование малых напряжений, например, не более 50 В переменного или 120 постоянного; электрическое разделение сетей на отдельные короткие участки; защитное заземление и зануление; применение средств индивидуальной защиты, таких как плакаты и знаки безопасности, изолирующие подставки, защитная одежда.

4.3.1.Экологическая безопасность

При использовании ПК могут возникнуть следующие виды негативного воздействия на окружающую среду:

- загрязнение почвы при утилизации старого ПК.

ПК после завершения использования (срока эксплуатации) можно отнести к отходам электронной промышленности. Переработка такого рода отходов осуществляется разделением на однородные компоненты. Существует два вида способов выделения компонентов: физические (механическая переработка, магнитная сепарация) и химические (пиролиз, металлургические методы). Эти способы используются для выделения пригодных для дальнейшего использования компонентов и направлением их

для дальнейшего использования: кремний, алюминий, золото, серебро, редкие металлы.

Для утилизации пластмассовых частей ПК могут применяться следующие способы: переработка отходов в полимерное сырьё и его повторное использование; сжигание вместе с другими отходами; пиролиз; захоронение на полигонах и на свалках.

ПК может содержать: тяжелые металлы, печатные платы с замедлителями горения, которые при горении могут выделять опасные диоксиды. Для опасных отходов используют теплоту сжигания, такой способ не исключает образования токсичных выбросов.

Отходы, которые не подлежат переработке, утилизации и вторичному использованию, подлежат захоронению на полигонах или в почве. Большое значение имеют нормативы предельно допустимых концентраций токсичных веществ в почве (ПДКп, мг/кг) в соответствии с [6].

3. Безопасность в чрезвычайных ситуациях

3.1. Пожарная безопасность

Пожар – неконтролируемое горение вне специального очага. Для реализации процесса горения необходимо наличие горючего, окислителя и источника воспламенения.

Офисное помещение (кабинет) по пожарной безопасности относится к категории В, в ней находятся горючие материалы и вещества в холодном состоянии [7]. По степени огнестойкости данное помещение относится к 3-й степени огнестойкости [8]. Возможные причины пожара: перегрузка в электросети, короткое замыкание, разрушение изоляции проводников.

Для локализации или ликвидации загорания на начальной стадии используются первичные средства пожаротушения: огнетушащие вещества (вода, песок, земля); огнетушащие материалы (грубошерстные куски материи – кошмы, асбестовые полотна, металлические сетки с малыми ячейками ит. п.); пожарный инвентарь (бочки и чаны с водой, пожарные ведра, ящики и песочницы с песком); пожарные краны на внутреннем водопроводе

противопожарного водоснабжения в сборе с пожарным стволом и пожарным рукавом; огнетушители [8].

Первичные средства пожаротушения обычно применяют до прибытия пожарной команды.

Здание должно соответствовать требованиям пожарной безопасности, а именно: иметь охранно-пожарную сигнализацию, план эвакуации, порошковые огнетушителей с поверенным клеймом, таблички с указанием направления к запасному (эвакуационному) выходу.

Углекислотные огнетушители ОУ-3 предназначены для тушения загораний веществ, горение которых не может происходить без доступа воздуха, загораний электроустановок, находящихся под напряжением не более 1000В, жидких и газообразных веществ (класс В, С).

Огнетушители не предназначены для тушения загорания веществ, горение которых может происходить без доступа воздуха (алюминий, магний и их сплавы, натрий, калий), такими огнетушителями нельзя тушить дерево.

В общественных зданиях и сооружениях на каждом этаже должно размещаться не менее двух переносных огнетушителей. Огнетушители следует располагать на видных местах вблизи от выходов из помещений на высоте не более 1,35 м. Размещение первичных средств пожаротушения в коридорах, переходах не должно препятствовать безопасной эвакуации людей.

4. Правовые и организационные вопросы обеспечения безопасности

Предъявляемые требования к расположению и компоновке рабочего места:

Высота рабочей поверхности стола для взрослых пользователей должна регулироваться в пределах (680÷800) мм, при отсутствии такой возможности высота рабочей поверхности стола должна составлять 725 мм [2].

Модульными размерами рабочей поверхности стола для ПК, на основании которых должны рассчитываться конструктивные размеры,

следует считать: ширину 800, 1000, 1200 и 1400 мм, глубину 800 и 1000 мм при нерегулируемой его высоте, равной 725 мм [2].

Рабочий стол должен иметь пространство для ног высотой не менее 600 мм, шириной – не менее 500 мм, глубиной на уровне колен – не менее 450мм и на уровне вытянутых ног – не менее 650 мм [2].

Конструкция рабочего стула должна обеспечивать:

– ширину и глубину поверхности сиденья не менее 400 мм;

– поверхность сиденья с закругленным передним краем;

– регулировку высоты поверхности сиденья в пределах (400÷550)мм и углам наклона вперед до 15 град, и назад до 5 град.;

– высоту опорной поверхности спинки (300±20) мм, ширину – не менее 380 мм и радиус кривизны горизонтальной плоскости –400 мм;

– угол наклона спинки в вертикальной плоскости в пределах ±30 градусов;

– стационарные или съемные подлокотники длиной не менее 250мм и шириной – (50÷70) мм;

– регулировку подлокотников по высоте над сиденьем в пределах(230±30) мм и внутреннего расстояния между подлокотниками в пределах (350÷500) мм [2].

Рабочее место пользователя ПК следует оборудовать подставкой для ног, имеющей ширину не менее 300 мм, глубину не менее 400 мм, регулировку по высоте в пределах до 150 мм и по углу наклона опорной поверхности подставки до 20°. Поверхность подставки должна быть рифленой и иметь по переднему краю бортик высотой 10 мм [2].

Клавиатуру следует располагать на поверхности стола на расстоянии (100÷300) мм от края, обращенного к пользователю или на специальной, регулируемой по высоте рабочей поверхности, отделенной от основной столешницы [2].

Экран видеомонитора должен находиться от глаз пользователя на расстоянии (600÷700 мм), но не ближе 500 мм [2].

Рабочее место с ПК должно располагаться по отношению к оконным проёмам так, чтобы свет падал слева. В качестве источников искусственного освещения должны использоваться люминесцентные лампы. Недопустим яркий не рассеянный верхний свет (с потолка). Сдерживать поток избыточного света от окон следует с помощью жалюзи (или тканевых штор). Влажную уборку помещения следует проводить ежедневно. Недопустима запыленность воздуха, пола, рабочей поверхности стола и техники. Помещение должно быть оборудовано системами вентиляции, кондиционирования и отопления. Запрещается работа на компьютере в подвальных помещениях.

В процессе испытаний может понадобиться присутствие специалиста на месте проведения испытаний. Для допуска специалиста, в чьи обязанности не входит работа в цехах и лабораториях, в которых проходят испытания, необходимо оформлять наряд-допуск к работам.

С работниками предприятий при поступлении на работу и дальнейшей деятельности проводятся инструктажи по охране труда.

Список использованных источников

1. СН 2.2.4/2.1.8.562 – 96. Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории застройки.
2. СанПиН 2.2.2/2.4.1340 – 03. Санитарно – эпидемиологические правила и нормативы «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы». – М.:Госкомсанэпиднадзор, 2003.
3. СанПиН 2.2.4.548 – 96. Гигиенические требования к микроклимату производственных помещений. М.: Минздрав России, 1997.
4. СП 52.13330.2016 Свод правил. Естественное и искусственное освещение.
5. СанПиН 2.2.4.548 – 96. Гигиенические требования к микроклимату производственных помещений. М.: Минздрав России, 1997.
6. ГН 2.1.7.2041 – 06. Предельно допустимые концентрации (ПДК) химических веществ в почве.
7. СП 12.13130.2009 Определение категорий помещений, зданий и наружных установок по взрывопожарной и пожарной опасности
8. Технический регламент «о требованиях пожарной безопасности» [Электронный ресурс]: Единая справочная служба Консорциума «Кодекс». – Режим доступа: свободный. Ссылка доступа: http://ezproxy.ha.tpu.ru:2065/docs/
9. Постановление правительства от 25 апреля 2012 года №390 « О противопожарном режиме»

Заключение

В ходе данной магистерской диссертации была исследована роботизированная мобильная платформа KUKA youBot; изучены основы кинематики и динамики данного робота, разработано программное обеспечение для управления роботом в симуляторе и в реальности; был создан алгоритм управления KUKA youBot в симуляторе V-REP; изучены основы систем технического зрения, в программе Matlab была создана математическая модель управления моментом на звеньях манипулятора.

СПИСОК ПУБЛИКАЦИЙ СТУДЕНТА

1. Ефименко А.П., Горисев С.А. Возможности CAM-системы Autodesk PowerMill 2018 в машиностроении // Технология машиностроения и материаловедение. – 2018 г. – № 2. С. 6–8.

2. Ефименко А.П., Сидорова А.А. Сравнение эффективности применения методов одномерной оптимизации // Научно-исследовательская работа студентов и аспирантов. –2018 г.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Symbiotic relationship between robots - a ROS ARDrone/YouBot library.
2. Redundancy modelling and resolution for robotic mobile manipulators: a general approach.
3. http://www.youbot-store.com/developers/kuka-youbot-kinematics-dynamics-and-3d-model-81
4. Constraint-Based Model Predictive Control for Holonomic Mobile Manipulators.
5. A Study of Accuracy and Time Delay for Bilateral Master-Slave Industrial Robotic Arm Manipulator System.
6. Developing of KUKA youBot Software for Education Process.
7. file:///C:/Users/Anastasia/Desktop/KUKA%20youbot%20статьи/cavallo2014.pdf
8. file:///C:/Users/Anastasia/Desktop/KUKA%20youbot%20статьи/10.1109@TCST.2017.2739706.pdf
9. Condition Monitoring and Cloud-based Energy Analysis for Autonomous Mobile Manipulation - Smart Factory Concept with LUHbots.
10. Investigation of Human-Robot Interface Performance in Household Environments.

Приложение А

(справочное)

Research of a robotic mobile platform KUKA youBot

Студент

Группа	ФИО	Подпись	Дата
8ЕМ61	Ефименко Анастасия Павловна		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор-консультант	Гончаров В. И.	д.т.н.		

Консультант-лингвист отделения иностранных языков ШБИП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель	Горбатова Т. Н.			

Literature review

Introduction

Industry in all its branches plays a very important role. In the modern world, a lot of attention is paid to the digitalization and robotization of production. Robotization is the main component of the automation of production. In practice, this process involves the use of robots and robotic systems in enterprises on an industrial scale. The tasks of recognizing objects and moving them, of movement along a given trajectory are very actual. Automated lines can be equipped with industrial robots, the presence of which will positively be reflected in the functioning of the whole complex of equipment. Also such mechanisms can be included in flexible automated production. The advantage of industrial robots is that for their reconfiguration to manufacture other products it does not require any special expenses, which ensures the process of product output is sufficiently versatile.

From above mentioned, robots play an important role in industry being an integral part of it, therefore it is necessary to organize their work correctly. KUKA youBot is a universal robotic platform, used for research purposes by students and a common representative of mobile robots on mecanum wheels for performing production tasks. The results of this work can be used for making laboratory works for further training of undergraduate students in the basics of mobile robotics.

The object of the research is the universal robotic mobile platform KUKA youBot, consisting of a platform on four mecanum wheels and a manipulator with five degrees of freedom (Figure 1).

Figure 1. Universal robotic mobile platform KUKA youBot

Basic information about KUKA youBot

KUKA youBot user manual

In the instruction for the user, I learned all the necessary basic information about the target of research. This source briefly describes the robot design and gives instructions for working with it.

The KUKA youBot base is an omni-directional mobile platform with four mecanum wheels. Figure 2 illustrates the attached base coordinate frame, as it will be used in the KUKA youBot API. It is located in the center of odometry. Positive values for rotation about the z axis result in a counterclockwise movement, as indicated by the blue arrow. The wheel numbering for the mecanum wheels is also shown.

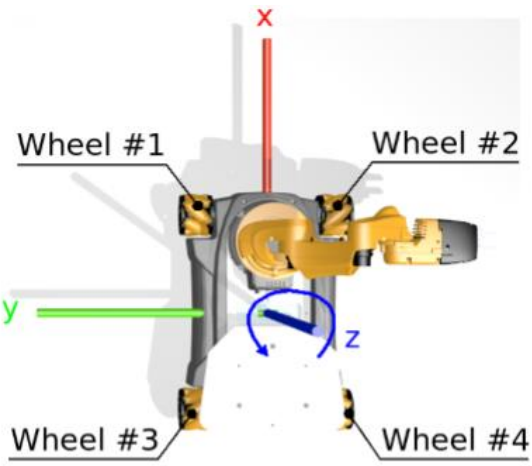


Figure 2: Overview of the KUKA youBot base. The Figure illustrates the attached base frame. Positive values for rotation about the z axis result in a counterclockwise movement, as indicated by the blue arrow.

The motor controllers of the four omni-directional wheels can be accessed via Ethernet and EtherCAT. The Ethernet cable can be either plugged into the onboard PC or to an external computer. The plug is a standard network cable plug that fits into each standard Ethernet port.

The KUKA youBot arm is a serial chain with five revolute axes. The end effector is a two-finger gripper, that can be removed. Figure 3 illustrates the basic kinematic structure. Similar as for the base the motor drivers for the individual joints can be accessed via Ethernet and EtherCAT.

Also describes the connection of KUKA youBot to a PC, the process of establishing programs for working with it, software for simple operations with KUKA youBot.

KUKA youBot software

Developing of KUKA youBot Software for Education Process

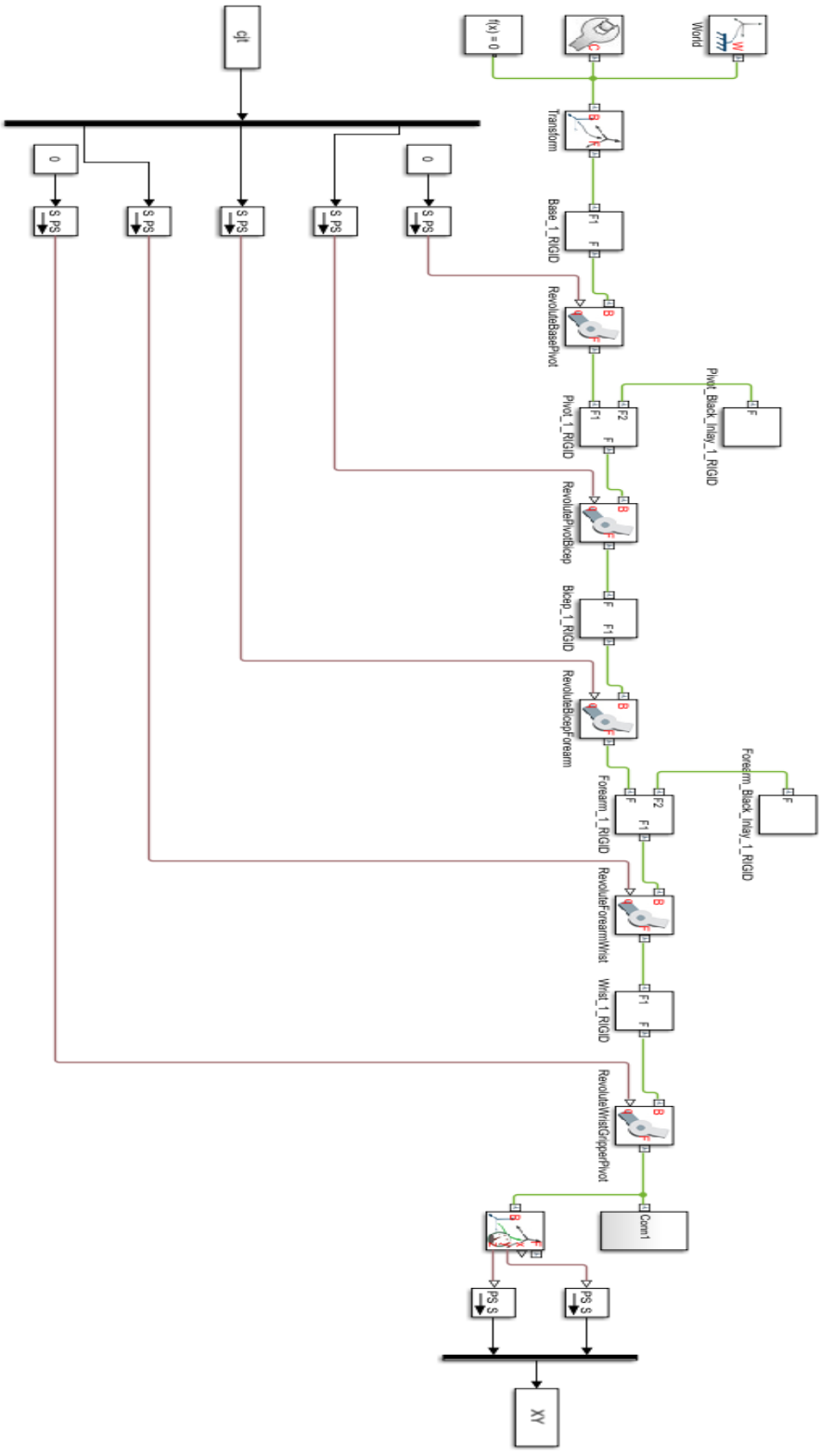
Nowadays there are many integrated development environments (IDE) for robot motion planning. Formally, from one side it could be graded into GUI features: visual programming environments and text-based programming environments. Again, each

group includes problem-solving environments and universal IDEs (that are support different models of robots). Neverthless, from other side, IDEs could be classified into functional features and implementation: educational and industrial IDEs.

For instance, IDE NXT-G (Center for Engineering Education and Outreach, 2013) and Trik Studio (KiberTech LLC, 2016) could be referred into visual educational IDEs. NXT-G is a graphical IDE, developed for popular customizable, programmable robots Lego Mindstorms NXT. It uses for NXT Brick module programming. This software has intuitive interface. Program developing looks like a timed sequence block diagram. Program flow determinates by instructions sequence. Trik Studio is commercial software oriented for education Trik robot programming. It includes simulator, that allows to test program without real robot system. In addition, there is a Microsoft Robotics Developer Studio (MRDS) (Microsoft, 2016). It could be referred into semi educational and industrial visual IDE. MRDS platform includes Visual Programming language (VPL) and visual simulated 3D environment. VPL recommended for newbies. It uses for programming strategies as block diagram. MRDS also support C# for professional developers. Among the educational problem-solving text-based environments are the most popular IDEs RobotC (Robomatter Inc., 2014) and BricxCC (BricxCC, 2011). RobotC is a leader among programming languages for robot competitions. It is based on C programming language, has no freeware version. There are five the most popular IDEs for industrial and science using: KUKA Sim Pro (KUKA AG, 2017), RobotStudio (ABB, 2017), MotoSim Touch (Yaskawa, 2017), V-REP (Coppelia Robotics GmbH, 2016), RoboDK (RoboDK, 2017). RobotStudio based on ABB VirtualController. It allows simulating with high level of reality, using real-time program and configuring models as well as the real manipulators. This IDE contains programming system MotoSim Touch and virtual robotic controller MotoSim EG-VRC. MotoSim Touch can switch between two modes: software (virtual) and hardware. KUKA Sim Pro is a software for 3D modelling with KUKA Robotics. It is a specific software for research centers and system integrators. There are many

components in standard library of KUKA Sim Pro. Intelligent components have grasp points that are allowed to connect their between themselves. KUKA Sim Pro supports API and COM for developing user's plugins. RoboDK is a powerful tool for autonomous programming and modeling complex processes. This software allows developing and simulating robot's behavior strategy of different manufacturers with Python language. In addition, there is a possibility of developing programs with special languages for different models.for instance. RoboDK is cross platform free software (Windows, Mac, Linux). There are API and C/C++, Python, Java, Lua, Matlab, Octave, Urbi support. The last of studied software is V-REP. Software developed for quick process modelling, verification, remote of robotic systems. It has distributed control architecture: each object/model can be controlled individually with own script, plugin, robot operating system (ROS) node, remote client's API. V-REP is a good solution for developing multi-robot control systems.

Приложение В



Система управления моментом

Листинг программного кода в симуляторе V-REP

```
function youbot()


  %% Initiate the connection to the simulator.


  disp('Program started');
  % Use the following line if you had to recompile remoteApi
  %vrep = remApi('remoteApi', 'extApi.h');
  vrep = remApi('remoteApi');
  vrep.simxFinish(-1);
  id = vrep.simxStart('127.0.0.1', 19997, true, true, 2000, 5);


  % If you get an error like:
  %   Remote API function call returned with error code: 64. Explanation: simxStart was not yet called.
  % Make sure your code is within a function! You cannot call V-REP from a script.


  if id < 0
    disp('Failed connecting to remote API server. Exiting.');
    vrep.delete();
    return;
  end
  fprintf('Connection %d to remote API server open.\n', id);


  % Make sure we close the connection whenever the script is interrupted.
  cleanupObj = onCleanup(@() cleanup_vrep(vrep, id));



  vrep.simxStartSimulation(id, vrep.simx_opmode_oneshot_wait);
```

```
% Retrieve all handles, and stream arm and wheel joints, the robot's pose, the Hokuyo, and the arm tip pose.
% The tip corresponds to the point between the two tongs of the gripper (for more details, see later or in the
% file focused/youbot_arm.m).
h = youbot_init(vrep, id);
h = youbot_hokuyo_init(vrep, h);

% Let a few cycles pass to make sure there's a value waiting for us next time we try to get a joint angle or
% the robot pose with the simx_opmode_buffer option.
pause(.2);

%% Youbot constants
% The time step the simulator is using
timestep = .05;

% Minimum and maximum angles for all joints. Only useful to implement custom IK.
armJointRanges = [-2.9496064186096, 2.9496064186096;
        -1.5707963705063, 1.308996796608;
        -2.2863812446594, 2.2863812446594;
        -1.7802357673645, 1.7802357673645;
        -1.5707963705063, 1.5707963705063 ];

% Definition of the starting pose of the arm (the angle to impose at each joint to be in the rest position).
startingJoints = [0, 30.91 * pi / 180, 52.42 * pi / 180, 72.68 * pi / 180, 0];

%% Preset values for the demo.
disp('Starting robot');

% Define the preset pickup pose for this demo.
pickupJoints = [90 * pi / 180, 19.6 * pi / 180, 113 * pi / 180, - 41 * pi / 180, 0];
```

```
% Parameters for controlling the youBot's wheels: at each iteration, those values will be set for the wheels.
% They are adapted at each iteration by the code.
forwBackVel = 0; % Move straight ahead.
rightVel = 0; % Go sideways.
rotateRightVel = 0; % Rotate.
prevOrientation = 0; % Previous angle to goal (easy way to have a condition on the robot's angular speed).
prevPosition = 0; % Previous distance to goal (easy way to have a condition on the robot's speed).

% Set the arm to its starting configuration.
res = vrep.simxPauseCommunication(id, true); % Send order to the simulator through vrep object.
vrchk(vrep, res); % Check the return value from the previous V-REP call (res) and exit in case of error.

for i = 1:5
    res = vrep.simxSetJointTargetPosition(id, h.armJoints(i), startingJoints(i), vrep.simx_opmode_oneshot);
    vrchk(vrep, res, true);
end

res = vrep.simxPauseCommunication(id, false);
vrchk(vrep, res);

% Initialise the plot.
plotData = true;
if plotData
    % Prepare the plot area to receive three plots: what the Hokuyo sees at the top (2D map), the point cloud and
    % the image of what is in front of the robot at the bottom.
    subplot(211);
    drawnow;
```

```
    % Create a 2D mesh of points, stored in the vectors X and Y. This will be used to display the area the robot can
    % see, by selecting the points within this mesh that are within the visibility range.
    [X, Y] = meshgrid(-5:.25:5, -5.5:.25:2.5); % Values selected for the area the robot will explore for this demo.
    X = reshape(X, 1, []); % Make a vector of the matrix X.
    Y = reshape(Y, 1, []);
  end

  % Make sure everything is settled before we start.
  pause(2);

  % Retrieve the position of the gripper.
  [res, homeGripperPosition] = vrep.simxGetObjectPosition(id, h.ptip, h.armRef, vrep.simx_opmode_buffer);
  vrchk(vrep, res, true);

  % Initialise the state machine.
  fsm = 'rotate';

  %% Start the demo.
  while true
    tic % See end of loop to see why it's useful.

    if vrep.simxGetConnectionId(id) == -1
      error('Lost connection to remote API.');
    end

    % Get the position and the orientation of the robot.
    [res, youbotPos] = vrep.simxGetObjectPosition(id, h.ref, -1, vrep.simx_opmode_buffer);
    vrchk(vrep, res, true);
    [res, youbotEuler] = vrep.simxGetObjectOrientation(id, h.ref, -1, vrep.simx_opmode_buffer);
```

```
vrchk(vrep, res, true);


%% Plot something if required.
if plotData
    % Read data from the depth sensor, more often called the Hokuyo (if you want to be more precise about
    % the way you control the sensor, see later for the details about this line or the file
    % focused/youbot_3dpointcloud.m).
    % This function returns the set of points the Hokuyo saw in pts. contacts indicates, for each point, if it
    % corresponds to an obstacle (the ray the Hokuyo sent was interrupted by an obstacle, and was not allowed to
    % go to infinity without being stopped).
    [pts, contacts] = youbot_hokuyo(vrep, h, vrep.simx_opmode_buffer);

    % Select the points in the mesh [X, Y] that are visible, as returned by the Hokuyo (it returns the area that
    % is visible, but the visualisation draws a series of points that are within this visible area).
    in = inpolygon(X, Y,...
            [h.hokuyo1Pos(1), pts(1, :), h.hokuyo2Pos(1)],...
            [h.hokuyo1Pos(2), pts(2, :), h.hokuyo2Pos(2)]);

    % Plot those points. Green dots: the visible area for the Hokuyo. Red starts: the obstacles. Red lines: the
    % visibility range from the Hokuyo sensor.
    % The youBot is indicated with two dots: the blue one corresponds to the rear, the red one to the Hokuyo
    % sensor position.
    subplot(211)
    plot(X(in), Y(in), '.g',...
        pts(1, contacts), pts(2, contacts), '*r',...
        [h.hokuyo1Pos(1), pts(1, :), h.hokuyo2Pos(1)], [h.hokuyo1Pos(2), pts(2, :), h.hokuyo2Pos(2)], 'r',...
        0, 0, 'ob',...
```

```
        h.hokuyo1Pos(1), h.hokuyo1Pos(2), 'or',...
        h.hokuyo2Pos(1), h.hokuyo2Pos(2), 'or');
    axis([-5.5, 5.5, -5.5, 2.5]);
    axis equal;
    drawnow;
end
angl = -pi/2;

%% Apply the state machine.
if strcmp(fsm, 'rotate')
    %% First, rotate the robot to go to one table.
    % The rotation velocity depends on the difference between the current angle and the target.
    rotateRightVel = angdiff(angl, youbotEuler(3));

    % When the rotation is done (with a sufficiently high precision), move on to the next state.
    if (abs(angdiff(angl, youbotEuler(3))) < .1 / 180 * pi) && ...
            (abs(angdiff(prevOrientation, youbotEuler(3))) < .01 / 180 * pi)
        rotateRightVel = 0;
        fsm = 'drive';
    end

    prevOrientation = youbotEuler(3);
elseif strcmp(fsm, 'drive')
    %% Then, make it move straight ahead until it reaches the table (x = 3.167 m).
    % The further the robot, the faster it drives. (Only check for the first dimension.)
    % For the project, you should not use a predefined value, but rather compute it from your map.
    forwBackVel = - (youbotPos(1) + 3.167);

    % If the robot is sufficiently close and its speed is sufficiently low, stop it and move its arm to
    % a specific location before moving on to the next state.
    if (youbotPos(1) + 3.167 < .001) && (abs(youbotPos(1) - prevPosition) < .001)
```

```
            forwBackVel = 0;

            % Change the orientation of the camera to focus on the table (preparation for next state).
            vrep.simxSetObjectOrientation(id, h.rgbdCasing, h.ref, [0, 0, pi/4], vrep.simx_opmode_oneshot);

            % Move the arm to the preset pose pickupJoints (only useful for this demo; you should compute it based
            % on the object to grasp).
            for i = 1:5
               res = vrep.simxSetJointTargetPosition(id, h.armJoints(i), pickupJoints(i),...
                                     vrep.simx_opmode_oneshot);
               vrchk(vrep, res, true);
            end

            fsm = 'snapshot';
         end
         prevPosition = youbotPos(1);
      elseif strcmp(fsm, 'snapshot')
         %% Read data from the depth camera (Hokuyo)
         % Reading a 3D image costs a lot to VREP (it has to simulate the image). It also requires a lot of
         % bandwidth, and processing a 3D point cloud (for instance, to find one of the boxes or cylinders that
         % the robot has to grasp) will take a long time in MATLAB. In general, you will only want to capture a 3D
         % image at specific times, for instance when you believe you're facing one of the tables.

         % Reduce the view angle to pi/8 in order to better see the objects. Do it only once.
         % ^^^^^^    ^^^^^^^^^^   ^^^^                          ^^^^^^^^^^^^^^^^
         % simxSetFloatSignal                                simx_opmode_oneshot_wait
         %          |
         %          rgbd_sensor_scan_angle
```

```
% The depth camera has a limited number of rays that gather information. If this number is
concentrated
% on a smaller angle, the resolution is better. pi/8 has been determined by experimentation.
res = vrep.simxSetFloatSignal(id, 'rgbd_sensor_scan_angle', pi / 8,
vrep.simx_opmode_oneshot_wait);
vrchk(vrep, res);

% Ask the sensor to turn itself on, take A SINGLE POINT CLOUD, and turn itself off again.
% ^^^   ^^^^^^          ^^    ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
% simxSetIntegerSignal       1      simx_opmode_oneshot_wait
%       |
%       handle_xyz_sensor
res = vrep.simxSetIntegerSignal(id, 'handle_xyz_sensor', 1, vrep.simx_opmode_oneshot_wait);
vrchk(vrep, res);

% Then retrieve the last point cloud the depth sensor took.
% If you were to try to capture multiple images in a row, try other values than
% vrep.simx_opmode_oneshot_wait.
fprintf('Capturing a point cloud...\n');
pts = youbot_xyz_sensor(vrep, h, vrep.simx_opmode_oneshot_wait);
% Each column of pts has [x;y;z;distancetosensor]. However, plot3 does not have the same frame of
reference as
% the output data. To get a correct plot, you should invert the y and z dimensions.

% Here, we only keep points within 1 meter, to focus on the table.
pts = pts(1:3, pts(4, :) < 1);

if plotData
  subplot(223)
  plot3(pts(1, :), pts(3, :), pts(2, :), '*');
  axis equal;
  view([-169 -46]);
```

```
end

% Save the point cloud to pc.xyz. (This file can be displayed with http://www.meshlab.net/.)
fileID = fopen('pc.xyz','w');
fprintf(fileID,'%f %f %f\n', pts);
fclose(fileID);
fprintf('Read %i 3D points, saved to pc.xyz.\n', max(size(pts)));

%% Read data from the RGB camera
% This starts the robot's camera to take a 2D picture of what the robot can see.
% Reading an image costs a lot to VREP (it has to simulate the image). It also requires a lot of bandwidth,
% and processing an image will take a long time in MATLAB. In general, you will only want to capture
% an image at specific times, for instance when you believe you're facing one of the tables or a basket.

% Ask the sensor to turn itself on, take A SINGLE IMAGE, and turn itself off again.
% simxSetIntegerSignal        1       simx_opmode_oneshot_wait
%       handle_rgb_sensor
res = vrep.simxSetIntegerSignal(id, 'handle_rgb_sensor', 1, vrep.simx_opmode_oneshot_wait);
vrchk(vrep, res);

% Then retrieve the last picture the camera took. The image must be in RGB (not gray scale).

%     simxGetVisionSensorImage2     h.rgbSensor                 0
% If you were to try to capture multiple images in a row, try other values than
% vrep.simx_opmode_oneshot_wait.
fprintf('Capturing image...\n');
[res, resolution, image] = vrep.simxGetVisionSensorImage2(id, h.rgbSensor, 0, vrep.simx_opmode_oneshot_wait);
vrchk(vrep, res);
fprintf('Captured %i pixels (%i x %i).\n', resolution(1) * resolution(2), resolution(1), resolution(2));
```

```
    % Finally, show the image.
    if plotData
        subplot(224)
        imshow(image);
        drawnow;
    end

    % Next state.
    fsm = 'extend';
elseif strcmp(fsm, 'extend')
    %% Move the arm to face the object.
    % Get the arm position.
    [res, tpos] = vrep.simxGetObjectPosition(id, h.ptip, h.armRef, vrep.simx_opmode_buffer);
    vrchk(vrep, res, true);

    % If the arm has reached the wanted position, move on to the next state.
    % Once again, your code should compute this based on the object to grasp.
    if norm(tpos - [0.3259, -0.0010, 0.2951]) < .002
        % Set the inverse kinematics (IK) mode to position AND orientation.
        res = vrep.simxSetIntegerSignal(id, 'km_mode', 2, vrep.simx_opmode_oneshot_wait);
        vrchk(vrep, res, true);
        fsm = 'reachout';
    end
elseif strcmp(fsm, 'reachout')
    %% Move the gripper tip along a line so that it faces the object with the right angle.
    % Get the arm tip position. The arm is driven only by the position of the tip, not by the angles of
    % the joints, except if IK is disabled.
    % Following the line ensures the arm attacks the object with the right angle.
    [res, tpos] = vrep.simxGetObjectPosition(id, h.ptip, h.armRef, vrep.simx_opmode_buffer);
    vrchk(vrep, res, true);
```

```
        % If the tip is at the right position, go on to the next state. Again, this value should be computed
based
        % on the object to grasp and on the robot's position.
        if tpos(1) > .39
            fsm = 'grasp';
        end

        % Move the tip to the next position along the line.
        tpos(1) = tpos(1) + .01;
        res = vrep.simxSetObjectPosition(id, h.ptarget, h.armRef, tpos, vrep.simx_opmode_oneshot);
        vrchk(vrep, res, true);
    elseif strcmp(fsm, 'grasp')
        %% Grasp the object by closing the gripper on it.
        % Close the gripper. Please pay attention that it is not possible to adjust the force to apply:
        % the object will sometimes slip from the gripper!
        res = vrep.simxSetIntegerSignal(id, 'gripper_open', 0, vrep.simx_opmode_oneshot_wait);
        vrchk(vrep, res);

        % Make MATLAB wait for the gripper to be closed. This value was determined by experiments.
        pause(2);

        % Disable IK; this is used at the next state to move the joints manually.
        res = vrep.simxSetIntegerSignal(id, 'km_mode', 0, vrep.simx_opmode_oneshot_wait);
        vrchk(vrep, res);
        fsm = 'backoff';
    elseif strcmp(fsm, 'backoff')
        %% Go back to rest position.
        % Set each joint to their original angle, as given by startingJoints. Please note that this operation is
not
        % instantaneous, it takes a few iterations of the code for the arm to reach the requested pose.
        for i = 1:5
```

```
            res = vrep.simxSetJointTargetPosition(id, h.armJoints(i), startingJoints(i),
vrep.simx_opmode_oneshot);
            vrchk(vrep, res, true);
        end

        % Get the gripper position and check whether it is at destination (the original position).
        [res, tpos] = vrep.simxGetObjectPosition(id, h.ptip, h.armRef, vrep.simx_opmode_buffer);
        vrchk(vrep, res, true);
        if norm(tpos - homeGripperPosition) < .02
            % Open the gripper when the arm is above its base.
            res = vrep.simxSetIntegerSignal(id, 'gripper_open', 1, vrep.simx_opmode_oneshot_wait);
            vrchk(vrep, res);
        end

        if norm(tpos - homeGripperPosition) < .002
            fsm = 'finished';
        end
    elseif strcmp(fsm, 'finished')
        %% Demo done: exit the function.
        pause(3);
        break;
    else
        error('Unknown state %s.', fsm);
    end

    % Update wheel velocities using the global values (whatever the state is).
    h = youbot_drive(vrep, h, forwBackVel, rightVel, rotateRightVel);

    % Make sure that we do not go faster than the physics simulation (each iteration must take roughly 50
ms).
    elapsed = toc;
    timeleft = timestep - elapsed;
```

```
        if timeleft > 0
            pause(min(timeleft, .01));
        end
    end

end % main function
```

Управление траекторией

 **/

#include <youbot_trajectory_action_server/joint_trajectory_action.h>

#include <youbot_trajectory_action_server/joint_state_observer.h>

#include <brics_actuator/JointPositions.h>

#include <brics_actuator/JointVelocities.h>

#include <kdl/trajectory_composite.hpp>

#include <kdl/trajectory_segment.hpp>

#include <kdl/velocityprofile_spline.hpp>

#include <kdl/path_line.hpp>

#include <kdl/rotational_interpolation_sa.hpp>

#include <boost/units/systems/angle/degrees.hpp>

#include <boost/units/conversion.hpp>

#include <boost/units/systems/si/length.hpp>

#include <boost/units/systems/si/plane_angle.hpp>

#include <boost/units/systems/si/angular_velocity.hpp>

#include <control_msgs/FollowJointTrajectoryActionResult.h>

JointTrajectoryAction::JointTrajectoryAction(JointStateObserver* jointStateObserver) :
jointStateObserver(jointStateObserver)

{

```
    setPositionGain(5.0);
    setVelocityGain(0.0);
    setFrequency(50);

}

JointTrajectoryAction::JointTrajectoryAction(JointStateObserver* jointStateObserver,
                                double positionGain,
                                double velocityGain,
                                double frequency) : jointStateObserver(jointStateObserver)
{



    setFrequency(frequency);
    setPositionGain(positionGain);
    setVelocityGain(velocityGain);

}

JointTrajectoryAction::JointTrajectoryAction(const JointTrajectoryAction& orig) :
jointStateObserver(orig.jointStateObserver)
{

    setPositionGain(orig.getPositionGain());
    setVelocityGain(orig.getVelocityGain());
    setFrequency(orig.getFrequency());
}

JointTrajectoryAction::~JointTrajectoryAction()
{
```

```
}

void JointTrajectoryAction::setFrequency(double frequency)
{
   this->frequency = frequency;
}

double JointTrajectoryAction::getFrequency() const
{
   return frequency;
}

void JointTrajectoryAction::setVelocityGain(double velocityGain)
{
   this->velocityGain = velocityGain;
}

double JointTrajectoryAction::getVelocityGain() const
{
   return velocityGain;
}

void JointTrajectoryAction::setPositionGain(double positionGain)
{
   this->positionGain = positionGain;
}

double JointTrajectoryAction::getPositionGain() const
{
   return positionGain;
}
```

```
double JointTrajectoryAction::calculateVelocity(double actualAngle,
                                double actualVelocity,
                                const KDL::Trajectory_Composite& trajectoryComposite,
                                double elapsedTimeInSec)
{

   double error = 0;

   if (trajectoryComposite.Duration() > 0 && elapsedTimeInSec <= trajectoryComposite.Duration())
   {
      double actualTime = elapsedTimeInSec;

      double desiredAngle = trajectoryComposite.Pos(actualTime).p.x();
      double desiredVelocity = trajectoryComposite.Vel(actualTime).vel.x();
      double velocityError = desiredVelocity - actualVelocity;
      double positionError = desiredAngle - actualAngle;
      double gain1 = getPositionGain();
      double gain2 = getVelocityGain();

      error = gain1 * positionError + gain2 * velocityError;

   }

   return error;
}

void JointTrajectoryAction::controlLoop(const std::vector<double>& actualJointAngles,
                           const std::vector<double>& actualJointVelocities,
                           const KDL::Trajectory_Composite* trajectoryComposite,
                           int numberOfJoints,
```

```
                        ros::Time startTime,
                        std::vector<double>& velocities)
{

   velocities.clear();
   double elapsedTime = ros::Duration(ros::Time::now() - startTime).toSec();

   for (int i = 0; i < numberOfJoints; ++i)
   {
      double velocity = calculateVelocity(actualJointAngles[i],
                           actualJointVelocities[i],
                           trajectoryComposite[i],
                           elapsedTime);

      velocities.push_back(velocity);

   }

}

void JointTrajectoryAction::setTargetTrajectory(double angle1,
                              double angle2,
                              double duration,
                              KDL::Trajectory_Composite& trajectoryComposite)
{

   KDL::Frame pose1(KDL::Rotation::RPY(0, 0, 0), KDL::Vector(angle1, 0, 0));
   KDL::Frame pose2(KDL::Rotation::RPY(0, 0, 0), KDL::Vector(angle2, 0, 0));

   KDL::Path_Line* path = new KDL::Path_Line(pose1, pose2, new KDL::RotationalInterpolation_SingleAxis(),
0.001);
```

```
    KDL::VelocityProfile_Spline* velprof = new KDL::VelocityProfile_Spline();

    velprof->SetProfileDuration(0, path->PathLength(),
                                duration);

    KDL::Trajectory_Segment* trajectorySegment = new KDL::Trajectory_Segment(path, velprof);
    trajectoryComposite.Add(trajectorySegment);
}

void JointTrajectoryAction::execute(const control_msgs::FollowJointTrajectoryGoalConstPtr& goal, Server*
as)
{

    current_state.name = goal->trajectory.joint_names;
    current_state.position.resize(current_state.name.size());
    current_state.velocity.resize(current_state.name.size());
    current_state.effort.resize(current_state.name.size());

    sensor_msgs::JointState angle1;
    angle1.name = goal->trajectory.joint_names;
    angle1.position.resize(angle1.name.size());
    angle1.velocity.resize(angle1.name.size());
    angle1.effort.resize(angle1.name.size());

    sensor_msgs::JointState angle2;
    angle2.name = goal->trajectory.joint_names;
    angle2.position.resize(angle2.name.size());
    angle2.velocity.resize(angle2.name.size());
    angle2.effort.resize(angle2.name.size());

    const uint numberOfJoints = current_state.name.size();
```

```
KDL::Trajectory_Composite trajectoryComposite[numberOfJoints];

angle2.position = goal->trajectory.points.at(0).positions;
for (uint i = 1; i < goal->trajectory.points.size(); i++)
{
   angle1.position = angle2.position;
   angle2.position = goal->trajectory.points.at(i).positions;

   double duration = (goal->trajectory.points.at(i).time_from_start -
         goal->trajectory.points.at(i - 1).time_from_start).toSec();

   for (uint j = 0; j < numberOfJoints; ++j)
   {
      setTargetTrajectory(angle1.position.at(j),
                  angle2.position.at(j),
                  duration,
                  trajectoryComposite[j]);

   }
}

const double dt = 1.0 / getFrequency();
std::vector<double> velocities;
ros::Time startTime = ros::Time::now();

for (double time = 0; time <= trajectoryComposite[0].Duration(); time = time + dt)
{

   controlLoop(current_state.position,
            current_state.velocity,
            trajectoryComposite,
```

```
            numberOfJoints,
            startTime,
            velocities);

    brics_actuator::JointVelocities command;
    std::vector <brics_actuator::JointValue> armJointVelocities;
    armJointVelocities.resize(current_state.name.size());

    for (uint j = 0; j < numberOfJoints; j++)
    {
        armJointVelocities[j].joint_uri = current_state.name.at(j);
        armJointVelocities[j].value = velocities[j];
        armJointVelocities[j].unit = boost::units::to_string(boost::units::si::radian_per_second);
    }

    command.velocities = armJointVelocities;
    jointStateObserver->updateVelocity(command);

    ros::Duration(dt).sleep();
}

sensor_msgs::JointState goal_state;
goal_state.name = goal->trajectory.joint_names;
goal_state.position = goal->trajectory.points.back().positions;

brics_actuator::JointPositions command;
std::vector <brics_actuator::JointValue> armJointPositions;
armJointPositions.resize(current_state.name.size());

for (uint j = 0; j < numberOfJoints; j++)
{
```

```
    armJointPositions[j].joint_uri = current_state.name.at(j);
    armJointPositions[j].value = goal_state.position.at(j);
    armJointPositions[j].unit = boost::units::to_string(boost::units::si::radian);
  }

  command.positions = armJointPositions;
  jointStateObserver->updatePosition(command);

  // now wait until completion
  double joint_error_tolerance = 0.01; // no idea if that is too much
  bool finished = false;

  control_msgs::FollowJointTrajectoryResult result;
  result.error_code = control_msgs::FollowJointTrajectoryResult::SUCCESSFUL;
  as->setSucceeded(result);
  return;
}

void JointTrajectoryAction::jointStateCallback(const sensor_msgs::JointState& joint_state)
{
  int k = current_state.name.size();

  if (k != 0)
  {
    for (uint i = 0; i < joint_state.name.size(); i++)
    {
      for (uint j = 0; j < current_state.name.size(); j++)
      {
        if (current_state.name.at(j) == joint_state.name.at(i))
        {
          current_state.position[j] = joint_state.position.at(i);
```

```
                    current_state.velocity[j] = joint_state.velocity.at(i);
                k--;
                break;
              }
            }
            if (k == 0)
            {
              break;
            }
          }
        }
      }
```